Calhoun: The NPS Institutional Archive

DSpace Repository

Theses and Dissertations 1. Thesis and Dissertation Collection, all items

1969

Computer processing for display of three-dimensional structures.

Desens, Robert Bruce

http://hdl.handle.net/10945/24632

This publication is a work of the U.S. Government as defined in Title 17, United States Code, Section 101. Copyright protection is not available for this work in the United States.

Downloaded from NPS Archive: Calhoun

Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

http://www.nps.edu/library

NPS ARCHIVE
1969
DESENS, R.

COMPUTER PROCESSING FOR DISPLAY OF THREE-DIMENSIONAL STRUCTURES

by

Robert Bruce Desens

DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CA 93943-5101

United States
Naval Postgraduate School

THESIS

COMPUTER PROCESSING FOR DISPLAY OF THREE-DIMENSIONAL STRUCTURES

by

Robert Bruce Desens

October 1969

T-132,283

This document has been approved for public release and sale; its distribution is unlimited.

LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIF. 93940

Computer Processing for Display of Three-Dimensional Structures

by

Robert Bruce Desens
Lieutenant, United States Navy
B.S.E.E., Purdue University, 1961

Submitted in partial fulfillment of the
requirements for the degree of

ELECTRICAL ENGINEER

from the

NAVAL POSTGRADUATE SCHOOL
October 1969

PS ARCHIVE
969
ESENS, R.

ABSTRACT

The field of computer graphics applied to three-dimensional space is introduced through a discussion of perspective transformations, data structure, contour lines, and the problem of hidden-line removal. The transformation of three-dimensional coordinates into two-dimensional picture-plane coordinates is developed for twelve degrees of freedom, allowing the simultaneous movement and rotation of both the object under view and the observer. Basic concepts and requirements for the structure of data and ideas for the use of contour lines are discussed as a relative part of the field of three-dimensional computer graphics. An algorithm for the removal of hidden lines is explained for the case where the objects under view can be assumed to be constructed of bounded plane surfaces.

DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CA 93943-5101

TABLE OF CONTENTS

I. INTRODUCTION 13
  A. THE PICTURE COMPLEXITY 13
  B. TYPES OF THREE-DIMENSIONAL DISPLAY 13
    1. Pseudo 3-D Display 14
    2. Stereoscopic 3-D Display 14
    3. Volumetric 3-D Display 14
  C. IMPLEMENTATION 14
    1. The Basic Transformation 15
    2. Extension to Include Viewing Plane Movement 15
    3. Storage and Access Requirements 15
    4. Contour Lines 16
    5. Hidden-Line Removal 16
    6. Summary 16
    7. Appendixes 17
II. THREE-DIMENSIONAL PERSPECTIVE TRANSFORMATION 18
  A. THE AXES 18
    1. The Reference Axes 19
    2. The Object Axes 19
    3. Viewing Plane Axes 21
  B. PERSPECTIVE TRANSFORMATION 21
  C. OFFSET 23
  D. SCALE 23
  E. ROTATION 23
    1. Euler Angles 24

2. Fixed-Axes Rotation 25

3. Choice of Rotation 26

F. HOMOGENEOUS TRANSFORMATION MATRIX 26

G. DISPLAY COORDINATES 29

III. MOVEMENT OF THE VIEWING PLANE 30

A. THE HOMOGENEOUS MATRIX 30

B. TRANSLATION OF THE VIEWING PLANE 31

C. ORIENTATION OF THE VIEWING PLANE 33

D. SUMMARY 34

IV. DATA STRUCTURE 35

A. BASIC STRUCTURES 35

1. Structuring for Points Only 35

2. Structuring for Connected Lines 37

3. Structuring for Line Segments 37

4. Structuring for Simple Three-Dimensional Figures 38

5. Increased Complexity 40

B. STRUCTURE FOR HIDDEN-LINE REMOVAL 45

1. Node (Point) Information 47

2. Faces (Surface or Polygon) Information 48

3. Edge (Line or Segment) Information 51

C. APPLICATION DEPENDENCE 51

V. CONTOUR LINES 54

A. DEFINITIONS 54

B. ENCODING CONTOUR LINES 55

C. OBTAINING CONTOUR LINES 57

D. DISPLAY OF CONTOURS 58

1. Determination of Limits of Visibility 58

2. Variations 61

E. FURTHER STUDY 61

VI. HIDDEN-LINE REMOVAL 62

A. THE PREVIOUS REPORTS 62

1. Quantitative Invisibility 63

2. Invisibility by Sets 63

B. DEFINITIONS 64

C. SPECIAL TESTS 65

1. Implied Vorticity 65

2. Exterior-Interior Determination 70

3. Intersection of a Plane and a Line (PIERC) 72

D. ALGORITHMS 74

1. General Program 75

2. FACES (Subroutine for Entering Data) 79

3. Connecting the Defining Points (LINES) 81

4. Detection of Concave Dihedrals (CCVCVX) 82

5. Transformation to Perspective (CONV3D) 85

6. Suppressing Surfaces Hidden by the Object's Own Volume (HDNSRF) 90

7. Termination of the Picture in the Focal Plane (ALTER) 92

8. Subroutines Called During Segment Examination . . 94

a. Determination of the Nature of a Vertex (VRTNTR) 94

b. Determination of Losing Intersections (LNTRSC) 98

c. Determination of the Nature of a Near Point (TRTNNP) 104

d. Scissoring the Picture (WINDOW) 106

9. Examination of the Segments (CMPUT) 111

10. Preparation for a New View (ERASE) 115

E. SUMMARY 116

VII. SUMMARY 117

A. THE TRANSFORMATION 117

B. DATA STRUCTURE 118

C. HIDDEN-LINE ELIMINATION 120

D. THE COMPUTER OUTPUT 121

E. CONTOUR LINES 122

F. APPLICATIONS 122

G. DEVELOPMENT 124

1. Simulators Using Displays 124

2. On-Line Computer Graphics 125

3. On-Line Authorship 125

4. Extensions to Curved Surfaces 126

5. Tactical Displays 126

6. Display of Surfaces as a Function of Two Variables 126

H. FINAL 126

APPENDIX A 127

APPENDIX B 130

APPENDIX C 133

COMPUTER OUTPUT 145

COMPUTER PROGRAM (IBM 360-67) 192

COMPUTER PROGRAM (SDS 9300) 214

BIBLIOGRAPHY 232

INITIAL DISTRIBUTION LIST 236

FORM DD 1473 237

LIST OF FIGURES

1. Object Axes, Viewing-Plane Axes, and Reference Axes in Correspondence 20
2. Object Axes and Positive Rotations 20
3. Placement of Object Axes in Viewing Coordinate System 22
4. Relationships of Viewing Plane, Focal Point, and Offsets . . 22
5. Examples of Differing Data Structure Requirements 36
6. Data Structure for the Line 'A' 38
7. Wire-Frame Rectangular Block 39
8. Point List Data Array 41
9. Data Storage in Structure and Line Arrays 43
10. Arrangement of Storage Locations 44
11. Matrix Interpretation of Data Structure 46
12. Node Information 49
13. Faces Information 50
14. Lines Information 52
15. Freeman's Rectangular Grid 56
16. Encoded String 57
17. Contour Lines Examined for Left and Right Extremes 59
18. Implied Vorticity 65
19. Testing a Line for Intersection 67
20. Intersection Special Cases 68
21. Interior-Exterior Determination 72
22. Concave Dihedral 86
23. Test Segments Affecting Set of Blocking Faces 101

24. Intersection at an End Point of a Test Segment 101

25. Situation Requiring Determination of Closer Line Segment 103

26. Intersections in the Sighting Plane 103

27. The End-Point Ambiguity 105

28. Cropping the Picture to Fit the Frame 110

29. Special Cases of Multiple Intersections 114

30. Positions for Carrier Views (2 - 5) 123

31. Entering Lines 142

32. L-Block (1) 148

33. L-Block (2) 152

34. L-Block (3) 156

35. Carrier (1) 163

36. Carrier (2) 170

37. Carrier (3) 177

38. Carrier (4) 184

39. Carrier (5) 191

LIST OF TABLES

I. Parameters and Structures Data for L-Block (1) 145

II. Parameters and Structured Data for L-Block (2) 149

III. Parameters and Structured Data for L-Block (3) 153

IV. Parameters and Structured Data for Carrier (1) 157

V. Parameters and Structured Data for Carrier (2) 164

VI. Parameters and Structured Data for Carrier (3) 171

VII. Parameters and Structured Data for Carrier (4) 178

VIII. Parameters and Structured Data for Carrier (5) 185

LIST OF FLOWCHARTS

1. General Program 76
2. Subroutine FACES 80
3. Subroutine LINES 83
4. Subroutine CCVCVX 84
5. Subroutine CONV3D 87
6. Subroutine VMVMAT 88
7. Subroutine ROTATN and Subroutine FXRTAT 89
8. Subroutine HDNSRF 91
9. Subroutine ALTER 93
10. Subroutine VRTNTR 96
11. Subroutine PLNLIN and Subroutine PIERC 97
12. Subroutine LNTRSC 99
13. Subroutine TSTNNP 107
14. Subroutine WINDOW 108
15. Subroutine CMPUT 112

I. INTRODUCTION

Visualization in three-dimensional space is difficult for many people. If three-dimensional data is entered into a computer, and the computer can subsequently draw a perspective view from any given aspect, the visualization problem will have diminished to one of selecting the most convenient view or views.

A. THE PICTURE COMPLEXITY

A camera in taking a picture will furnish a perspective view of whatever lies in front of the lens within the limits of view. Similarly a computer can be made to draw a picture of an object about which dimensional information has been given.

The actual picture drawn is, of course, dependent on the input information, and detail will be reproduced accordingly. If enough detail is given--i.e. point by point--a picture quality reproduction would be possible. However, when one considers the number of points involved, it becomes apparent that the task for point-by-point processing is prohibitive in both storage and time.

On the other hand, if the information can be reduced to a simpler structure, the computer can handle the reproduction with much greater ease and considerably less memory.

B. TYPES OF THREE-DIMENSIONAL DISPLAYS

Several types of 3-D displays are possible. Characteristics of each type are discussed by Vlahos [Ref. 48] and are briefly summarized here.

1. Pseudo 3-D Display

Pseudo 3-D is a technique to give the illusion of three dimensions, but only monocular cues are given. The simplest of all pseudo 3-D displays is the perspective display. Addition of other cues such as shadow patterns, line size, motion cues, and brightness would add to the three-dimensional effect, but the display would remain a Pseudo 3-D display.

2. Stereoscopic 3-D Display

Contrasting with the Pseudo 3-D display is the Stereoscopic 3-D display. This technique employs binocular cues. Two images are formed, with only slight differences between them. The differences correspond to the different images that the right and left eyes would see. One of the most familiar techniques is to make the images of different colors or polarizations and then view the images through special glasses. The eyes then give the appropriate binocular effect.

3. Volumetric 3-D Display

In the Volumetric 3-D display, the display device itself is three-dimensional and for that reason Volumetric 3-D displays are limited in size. The span between the eyes in this case supplies the binocular cues. The information being displayed is itself in three dimensions.

C. IMPLEMENTATION

The assumption made for this study is that objects to be displayed will be represented by plane surfaces which are limited by their intersections with other plane surfaces. The computer operates on the describing data entered and computes the lines to be drawn for the Pseudo 3-D picture.

1. The Basic Transformation

The basic perspective transformation must occur on the points or nodes defining the object. Chapter II presents the basic three-dimensional transformation and includes a discussion of rotation and movement of the object, perspective transformation, offsets from the picture center, and a scale factor which is inversely proportional to magnification (where a scale factor of 1/2 corresponds to a magnification of 1).

2. Extension to Include Viewing Plane Movement

Chapter III extends the results of chapter II to include the movement of the viewing plane. The capability introduced here is to allow the independent movement and rotation of the object and viewing plane. The effect is similar to an aircraft flying around an aircraft carrier. The aircraft and ship's motion combine to introduce twelve degrees of freedom.

3. Storage and Access Requirements

Whenever data must be entered and stored in a computer to be utilized for some computation, a problem exists that generally has conflicting requirements, especially when large amounts of data are required. Storage in the computer should be kept to a minimum, but the data must be easily and preferably quickly available. Storage requirements may be reduced by placing or packing more than one piece of data in one computer word. Packing allows savings in storage but increases the time required to access the data. Chapter IV discusses a few basic data structures, finishing with a complex structure used to draw with the hidden line removed. The wire frame model is of course much simpler and quicker to draw than the object with hidden lines suppressed. Appel [Ref. 2] defines a ratio of the time to draw the wire frame model to the time to draw the model with hidden lines suppressed and suggests that this ratio is a good machine-independent measure of the efficiency of the hidden-line removal program. For assemblies of planes bounded by line segments and visibility determined by point-to-point examination, the time required to draw with hidden lines removed is from 100 to 1000 times the time required for the wire frame drawing. Appel was able to achieve ratios of from 10 to 20 with his method of quantitative invisibility. A definite ratio cannot be set for any method since this value will vary with the complexity of the drawing and the orientation of the view. The method of Galimberti and Montanari [Ref. 18] follows Appel's method with modifications, and should be slightly faster.

4. Contour Lines

A brief discussion of contour lines is presented in chapter V. Morse [Ref. 34] gives a formal mathematical presentation of contour lines and suggests that as methods for handling contour lines are developed, greater usage will ensue in the various branches of engineering.

5. Hidden-Line Removal

The hidden-line elemination is discussed in chapter VI using a combination of the methods of Galimberti and Montanari [Ref. 18] and Appel [Ref. 2] with modifications when they were desirable. The algorithm and implementation is discussed in this chapter with illustrations for various cases included.

6. Summary

A summary is found in chapter VIII with a brief discussion on program length and efficiency, as implemented on the SDS 9300 and the

IBM 360-67 located at the Naval Postgraduate School. Recommendations for future development are also found in this chapter.

7. Appendixes

Appendix A consists of a Glossary to aid the reader in following terminology and conventions developed throughout the various chapters.

Appendix B is a brief summary of programming changes necessary between the FORTRAN IV of the SDS 9300 and the FORTRAN IV of the IBM 360.

Appendix C contains instructions for the use of the program written to demonstrate the algorithms developed.

II. THREE-DIMENSIONAL PERSPECTIVE TRANSFORMATION

For the transformation of three-dimensional coordinates into pseudo 3-D coordinates, both digital and analog techniques have been used. The algorithms examined in this treatment of the subject are applicable to either or both methods of implementation. The general method as implemented in a machine-independent higher-level language is examined without regard to hardware. The transformation presented here is an adaptation and an extension of the transformation by L. G. Roberts [Ref. 29].

The desired degrees of freedom for the transformation include the rotation of the object, the movement of the object, the orientation of the viewing plane, and the movement of the viewing plane. This chapter will discuss the basic transformation for the rotation and movement of the object. The following chapter will extend the transformation to include the movement and orientation of the viewing plane. Motion of the viewing plane corresponds to motion of the observer. The transformation will utilize homogeneous coordinates.

A. THE AXES

There are three different right-hand orthogonal axes involved in the presentation which follows. These axes are the reference axes, the object axes, and the viewing-plane axes. A brief description of each of the sets of axes will help in understanding the chapter development.

1. The Reference Axes

All movement of the object axes and the viewing plane axes is in relation to a set of reference axes. The reference axes are aligned with the object axes and the viewing plane axes when no movement or rotation is present in either the object axes or the viewing plane axes. The reference axes are labeled X_{Ref}, Y_{Ref}, and Z_{Ref}. Figure 1 shows the three sets of axes in correspondence.

2. The Object Axes

Consider an object located in a coordinate frame with axes u, v, and w. Figure 2 shows this set of axes by itself, along with the directions of positive rotation.

Each object or group of objects is considered as if it had its own axes about which it must be rotated and which will hereafter be called the object axes to distinguish them from those of other coordinate systems. The order of rotation is important. All rotations are obtained by rotating two of the object axes about the third axis.

Rotation about the w axis is first and is measured by angle α. Axis u rotated into axis v defines the positive angle. This rotation is measured from the reference axes. Rotation about the v axis is second, and this is measured by β. The positive angle is defined by the direction of w rotated into u. This rotation is referenced to the rotated object axes. Last, rotation of v into w (about the u axis) is the positive angle γ. This rotation is also referenced to the rotated object axes. These angular rotations result in Euler angles similar to those described by Howe [Ref. 50]. The appropriate direction cosines are listed later in this chapter.

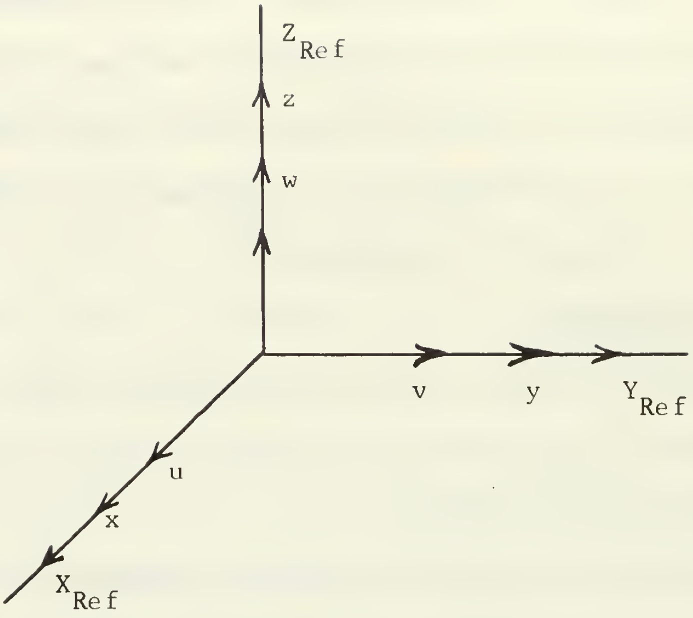


FIGURE 1

OBJECT AXES, VIEWING-PLANE AXES, AND REFERENCE AXES IN CORRESPONDENCE

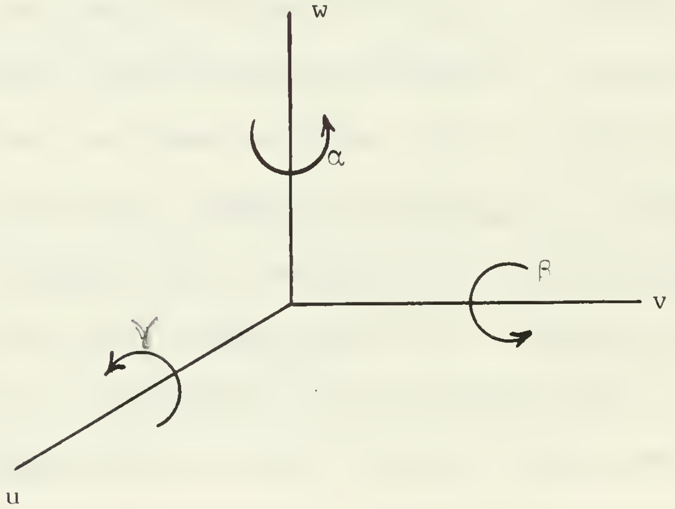


FIGURE 2

OBJECT AXES AND POSITIVE ROTATIONS

3. Viewing-Plane Axes

The object axes must now be placed in the coordinate system where it will be viewed. This coordinate system will be labeled x, y, and z. The relationship between the two systems is shown in Figure 3, with the reference axes corresponding in this case to the viewing-plane axes.

The viewing plane is the yz plane at $x = 0$, with limits of ± 1 in both y and z. Translation is measured in xyz coordinates from the xyz origin to the object axes origin. The translation coordinates will be called X_T, Y_T, and Z_T.

The viewer looks into the negative x half-space. The positive u axis extends from the origin of the object axes toward the viewer, when no rotation and no Y or Z translation is present.

B. PERSPECTIVE TRANSFORMATION

After rotation and translation, a perspective transformation is made. The focal point is located on the positive x axis. Roberts [Ref. 29] makes a comparison to a camera with one difference. The focal plane in this case is between the focal point and the object. The picture is square in this case, and is centered on the axis which also contains the focal point. Figure 4 shows a basic comparison. The ratio of the distance between the focal plane and the focal point to the distance between the center of the picture and one edge is approximately 3:1 or 4:1 for the standard camera lens (i.e. excluding telephoto and wide-angle lenses). For a telephoto effect, the distance between the focal point and the focal plane must be increased. Conversely, decreasing the distance yields a wide-angle effect.

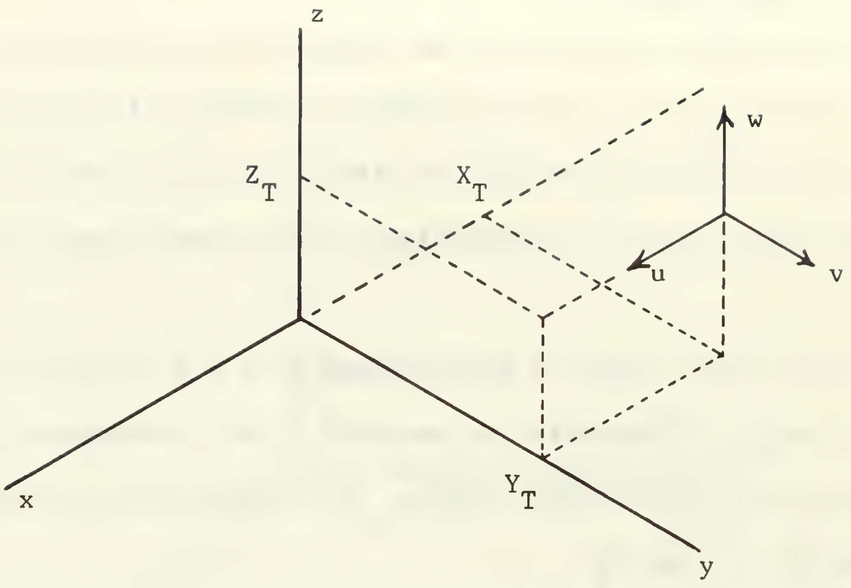


FIGURE 3

PLACEMENT OF OBJECT AXES IN VIEWING COORDINATE SYSTEM

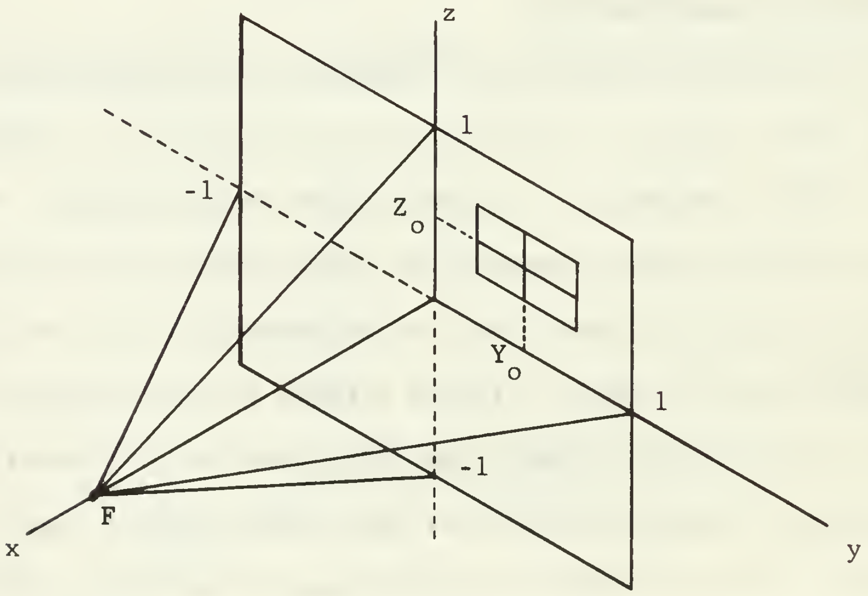


FIGURE 4

RELATIONSHIPS OF VIEWING PLANE, FOCAL POINT AND OFFSETS

C. OFFSET

It is possible, after taking a picture, to enlarge any section of it, centered anywhere in the picture. In the frame of reference used here, the distance from the center of the total picture to the center of the enlarged picture is measured as a y-offset and z-offset, called Y_o and Z_o. Figure 4 also shows the offset relationships.

D. SCALE

To control the magnification, another factor is introduced. This factor is called the scale. Scale is the normalized distance measured from the center to one edge of the picture. For a full size picture, scale is equal to 1/2. When an offset is introduced, scale should be decreased, giving an enlargement effect for that section of the picture. Standard cameras have a ratio of scale:focus in the range of 1:3 to 1:4.

Another way of looking at scale gives perhaps a better understanding of its function. The limits in the viewplane were set at $\pm$ 1, but when the limits are multiplied by scale for the usual case, an effective change in limits occurs, resulting in new limits of $\pm$ 0.5. This then gives a picture of a normalized size of 1 from top to bottom or from left to right. The corresponding angle for a focus located at $x = 2$ is $\pm$ 14 degrees vertically and horizontally from center.

The picture will always fill the entire focal plane area. If a value of 1/4 is chosen for scale, the result will be that the center of the picture is enlarged by a factor of two to fill the focal plane area.

E. ROTATION

In considering the rotation of the object, two basic choices of rotation are possible. The choice is between rotation employing Euler angles and rotation about fixed axes.

1. Euler Angles

Euler angles are employed when the coordinate axes rotate with the object. Angles used to measure yaw, pitch, and roll of an aircraft are an example of Euler angles. Pitch, for example, is always measured in the vertical plane which contains the longitudinal axis of the aircraft, and roll is measured about the longitudinal axis.

Modification of coordinates of a point due to Euler rotation can be found by the matrix

$$\begin{bmatrix} x_i & y_i & z_i \end{bmatrix} = \begin{bmatrix} u_i & v_i & w_i \end{bmatrix} \begin{bmatrix} \lambda_1 & \lambda_2 & \lambda_3 \\ \mu_1 & \mu_2 & \mu_3 \\ \nu_1 & \nu_2 & \nu_3 \end{bmatrix}$$

where (X_i, Y_i, Z_i) are the viewing plane coordinates,

(u_i, v_i, w_i) are the object axes coordinates,

and (λ, μ, ν) are the directional cosines for the angle that the uvw axes make with the xyz axes. Specifically, the direction cosines are

$$\lambda_1 = \cos\beta \cos\alpha$$

$$\lambda_2 = \cos\beta \sin\alpha$$

$$\lambda_3 = -\sin\beta$$

$$\mu_1 = -\cos\gamma \sin\alpha + \sin\gamma \sin\beta \cos\alpha$$

$$\mu_2 = \cos\alpha \cos\gamma + \sin\gamma \sin\beta \sin\alpha$$

$$\mu_3 = \sin\gamma \cos\beta$$

$$\nu_1 = \sin\gamma \sin\alpha + \cos\gamma \sin\beta \cos\alpha$$

$$\nu_2 = -\sin\gamma \cos\alpha + \cos\gamma \sin\beta \sin\alpha$$

$$\nu_3 = \cos\beta \cos\gamma$$

2. Fixed-Axes Rotation

If rotation is made about a specified inertial or reference axes, the rotation will be of the fixed type. Consider the picture window view of an object. If rotation is always made in reference to the picture window, the axes of rotation (the picture window and its perpendicular) are fixed.

The same matrix can be set up for fixed-axes rotation using direction cosines, but the direction cosines will be different in this case. The direction cosines can be found by rotation of each angle in turn.

$$\begin{bmatrix}\text{rotation}\\ \text{by}\\ \alpha\end{bmatrix}\begin{bmatrix}\text{rotation}\\ \text{by}\\ \beta\end{bmatrix}\begin{bmatrix}\text{rotation}\\ \text{by}\\ \gamma\end{bmatrix}$$

This is accomplished by

$$\begin{bmatrix}\cos\alpha & \sin\alpha & 0\\ -\sin\alpha & \cos\alpha & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\cos\beta & 0 & -\sin\beta\\ 0 & 1 & 0\\ \sin\beta & 0 & \cos\beta\end{bmatrix}\begin{bmatrix}1 & 0 & 0\\ 0 & \cos\gamma & \sin\gamma\\ 0 & -\sin\gamma & \cos\gamma\end{bmatrix}$$

which in turn simplifies to

$$\begin{bmatrix}\lambda_1 & \lambda_2 & \lambda_3\\ \mu_1 & \mu_2 & \mu_3\\ \nu_1 & \nu_2 & \nu_3\end{bmatrix}$$

where the direction cosines are now defined as

$\lambda_1 = \cos\alpha\cos\beta$

$\lambda_2 = \sin\alpha\cos\gamma + \sin\gamma\cos\alpha\sin\beta$

$\lambda_3 = \sin\alpha\sin\gamma - \cos\gamma\sin\beta\sin\alpha$

$\mu_1 = -\cos\beta\sin\alpha$

$\mu_2 = \cos\gamma\cos\alpha - \sin\gamma\sin\beta\sin\alpha$

$\alpha = \psi$
$\beta = \Theta$
$\gamma = \phi$

$\mu_3 = \sin\gamma \cos\alpha + \cos\gamma \sin\beta \sin\alpha$

$\nu_1 = \sin\beta$

$\nu_2 = -\sin\gamma \cos\beta$

$\nu_3 = \cos\gamma \cos\beta$

3. Choice of Rotation

Both types of rotation have been used for display. It was decided that for display of the object, the rotation utilizing Euler angles would be the easiest to visualize, provided that the axes or some indication of the axes could be displayed.

The fixed axes rotation is also needed and is used in another context. This will be discussed in the following chapter.

F. HOMOGENEOUS TRANSFORMATION MATRIX

By utilizing homogeneous coordinates, the functions of rotation, translation, perspective, offset and scale can be combined into one matrix. To accomplish this with the rotation matrix, a fourth coordinate is added to the three-coordinate system and can be considered as a variable scale factor. If the original coordinates are

$$(X\ Y\ Z),$$

the new coordinates are

$$(x_H\ y_H\ z_H\ w_H),$$

where

$$x_H = w_H X$$

$$y_H = w_H Y$$

$$z_H = w_H Z.$$

Of course if $w_H = 1$, then $x_H = X$, $y_H = Y$, and $z_H = Z$. The factor w_H may be any positive number, and can handily be chosen as a normalizing factor when using fixed-point rather than floating-point computers. The factor w_H can be changed for each set of coordinates defining a point since the factor cancels when converting to the display coordinates. Therefore when using a normalizing factor, each set of coordinates defining a point can be individually normalized. For floating-point operation, the factor is chosen as 1.0 to reduce the operations necessary.

Modifying the rotation matrix for the homogeneous system yields the following 4 x 4 matrix,

$$\begin{bmatrix} \lambda_1 & \lambda_2 & \lambda_3 & 0 \\ \mu_1 & \mu_2 & \mu_3 & 0 \\ \nu_1 & \nu_2 & \nu_3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

which reduces to the unit matrix if no rotation is present.

The translation matrix is simply

$$\begin{bmatrix} w_H & 0 & 0 & 0 \\ 0 & w_H & 0 & 0 \\ 0 & 0 & w_H & 0 \\ x_T & y_T & z_T & w_H \end{bmatrix} = w_H \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ X_T & Y_T & Z_T & 1 \end{bmatrix}$$

Premultiplying the translation matrix by the rotation matrix yields

$$w_H \begin{bmatrix} \lambda_1 & \lambda_2 & \lambda_3 & 0 \\ \mu_1 & \mu_2 & \mu_3 & 0 \\ \nu_1 & \nu_2 & \nu_3 & 0 \\ X_T & Y_T & Z_T & 1 \end{bmatrix}$$

The perspective transformation matrix is

$$\begin{bmatrix} F & 0 & 0 & -1 \\ 0 & F & 0 & 0 \\ 0 & 0 & F & 0 \\ 0 & 0 & 0 & F \end{bmatrix} = F \begin{bmatrix} 1 & 0 & 0 & -\frac{1}{F} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where F is the distance from the viewing plane (the yz-plane at $x = 0$) to the focal point.

The offset matrix is

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & Y_0 & Z_0 & 1 \end{bmatrix}$$

where Y_0 and Z_0 are the new centers of the picture.

The scale matrix is simply

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & S \end{bmatrix}$$

where S is the measure scale.

Combining all the functions into one matrix, thus

[ROTATION] [TRANSLATION] [PERSPECTIVE] [OFFSET] [SCALE]

yields the H transformation matrix, where

$$H = wF \begin{bmatrix} \lambda_1 & \lambda_2 - Y_0(\lambda_1)/F & \lambda_3 - Z_0(\lambda_1)/F & -S(\lambda_1)/F \\ \mu_1 & \mu_2 - Y_0(\mu_1)/F & \mu_3 - Z_0(\mu_1)/F & -S(\mu_1)/F \\ \nu_1 & \nu_2 - Y_0(\nu_1)/F & \nu_3 - Z_0(\nu_1)/F & -S(\nu_1)/F \\ X_T & Y_T - Y_0(X_T - F)/F & Z_T - Z_0(X_T - F)/F & -S(X_T - F)/F \end{bmatrix}$$

When a point is to be transformed, it must be expressed in homogeneous coordinates. Thus the point $(X_i\ Y_i\ Z_i)$ becomes $(x_i\ y_i\ z_i\ w_H)$. The transformation is then made by post-multiplying the point matrix by the H transformation matrix. Thus,

$$\begin{bmatrix} x_1 & y_1 & z_1 & w_H \end{bmatrix} \begin{bmatrix} H \end{bmatrix} = \begin{bmatrix} x_1' & y_1' & z_1' & w_H' \end{bmatrix}$$

where the primes now denote the complete perspective transformation.

G. DISPLAY COORDINATES

To obtain the display coordinates, division of the homogeneous transformed coordinates must be made by the w_H' coordinate. Thus,

$$Y_1' = y_1'/w_H'$$

$$Z_1' = z_1'/w_H'$$

where Y_1' is the horizontal displacement and Z_1' is the vertical displacement. The coordinate x_1' is an indication of depth and can be used for depth cues.

Since the factor w_HF appears in every homogeneous transformed coordinate, it will eventually be cancelled in arriving at display coordinates. There is no change if the factor is set to 1.0 in these transformations, and computing time can be reduced by eliminating several operations.

III. MOVEMENT OF THE VIEWING PLANE

Previously, movement of the object axes utilizing both rotation and translation was accomplished. Related to the movement of the object axes is the movement of the viewing plane to some arbitrary point and the orientation of the viewing plane to some arbitrary direction. The movement of the viewing-plane axes corresponds to motion of an observer. An aircraft flying around a fixed object and viewing the object from the various sides is an example of this capability.

It can be seen that the two situations can be thought of as the same, but with a change in reference. One might argue that there is really no difference and therefore unworthy of consideration as a separate problem. However, if one considers that possibly both the object and the viewing plane could move, a much larger set of applications is possible.

Consider again the aircraft as an example. This time let the aircraft fly around an aircraft carrier. The carrier has motion about each of its three object (ship's) axes. The aircraft, which contains the viewing plane, is moving about some inertial reference. The viewing plane therefore is moving in respect to a stationary point in some inertial frame.

A. THE HOMOGENEOUS MATRIX

Implementation of the movement in each of the three dimensions can be easily accomplished by retaining and expanding the previously developed matrices.

For illustrative purposes, write the matrix transformation already derived as

$$\begin{bmatrix} \lambda_1 & \lambda_2 & \lambda_3 & 0 \\ \mu_1 & \mu_2 & \mu_3 & 0 \\ \nu_1 & \nu_2 & \nu_3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ X_T & Y_T & Z_T & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -\frac{1}{F} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & Y_0 & Z_0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & S \end{bmatrix}$$

Combining the last three matrices yields

$$H = \begin{bmatrix} \lambda_1 & \lambda_2 & \lambda_3 & 0 \\ \mu_1 & \mu_2 & \mu_3 & 0 \\ \nu_1 & \nu_2 & \nu_3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ X_T & Y_T & Z_T & 1 \end{bmatrix} \begin{bmatrix} 1 & -Y_0/F & -Z_0/F & -S/F \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & Y_0 & Z_0 & S \end{bmatrix}$$

It is now necessary to have agreement on whether the translation of the viewing plane or the orientation of the viewing plane comes first. Translating the viewing plane to some point and them orienting it in the direction desired was considered to be easier to visualize than first orienting the view-plane and then translating it. The order of rotation when re-orienting the view-plane must also be specified. The same convention that was established with the object axes will be used with the viewing plane axes. Rotation (yaw) about the z axis is therefore measured first by the angle PSI. Pitch about the y axis is measured next by the angle THETA. Roll about the x axis is measured last by the angle PHI.

B. TRANSLATION OF THE VIEWING PLANE

Translation of the viewing plane is really the same as a negative translation by the same amount of the object axes. Let the object axes translation from the reference axes be

$$X_{OB} \; Y_{OB} \; Z_{OB}$$

and let the viewing plane translation from the reference axes be

$$X_{VP} \; Y_{VP} \; Z_{VP}.$$

Let the resultant translation be

$$X_T \; Y_T \; Z_T.$$

Then

$$H = \begin{bmatrix} \text{Rotation} \\ \text{Trans-} \\ \text{formation} \end{bmatrix} \underset{\substack{\text{Object-Axes} \\ \text{Translation}}}{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ X_{OB} & Y_{OB} & Z_{OB} & 1 \end{bmatrix}} \underset{\substack{\text{Viewing-Plane} \\ \text{Translation}}}{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -X_{VP} & -Y_{VP} & -Z_{VP} & 1 \end{bmatrix}} \begin{bmatrix} \text{Perspective} \\ \text{Offset} \\ \text{Scale} \\ \text{Trans-} \\ \text{formation} \end{bmatrix}$$

$$H = \begin{bmatrix} \text{Rotation} \\ \text{Trans-} \\ \text{formation} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ X_T & Y_T & Z_T & 1 \end{bmatrix} \begin{bmatrix} \text{Perspective} \\ \text{Offset} \\ \text{Scale} \\ \text{Transformation} \end{bmatrix}$$

where the resultant translation $(X_T \; Y_T \; Z_T)$ is

$$X_T = X_{OB} - X_{VP}$$

$$Y_T = Y_{OB} - Y_{VP}$$

$$Z_T = Z_{OB} - Z_{VP}.$$

Let a point being transformed be represented by the vector $\bar{P}$, where

$$\bar{P} = \begin{bmatrix} x_H & y_H & z_H & w_H \end{bmatrix}.$$

Then

$$\begin{bmatrix} \bar{P} \end{bmatrix} \begin{bmatrix} \text{Rotation} \end{bmatrix}$$

results in the coordinates of the points after rotation, expressed in viewing-plane coordinate values (corresponding to the reference plane since the viewing plane has not yet been moved). Call the resultant $\overline{P}'$. Then translating both the object axes and the viewing-plane axes

$$\begin{bmatrix} \overline{P}' \end{bmatrix} \begin{bmatrix} \text{Translation} \\ \text{Object Axes} \end{bmatrix} \begin{bmatrix} \text{Translation} \\ \text{View-Plane} \\ \text{Axes} \end{bmatrix}$$

results in the coordinates of the point after rotation and translation of both the object axes and the viewing-plane axes, expressed again in the new viewing-plane coordinates. Call this resultant $\overline{P}''$. At this point, of course, the reference axes and the viewing-plane axes no longer correspond.

C. ORIENTATION OF THE VIEW-PLANE

Re-orientation of the view-plane at this point can be looked upon as a rotation of a new object axes, say u' v' w'. These new object axes correspond to the viewing-plane axes before rotation and the coordinates $\overline{P}''$ which were found in the previous steps. If the axes did not correspond at this point, it would be necessary to evaluate new translations brought about by the reorientation of the viewing plane. In this example however, the axes were made to correspond.

Rotation of the axes now is a simple matter. Rotation of the axes is the negative of the viewing-plane rotation. However, the rotation of the object axes in this context must be around a fixed set of axes, the fixed axes corresponding to the axes of the viewing plane. If rotation utilizing Euler angles were used, the second rotation would be about the rotated object axes instead of the new viewing-plane axes.

After multiplying by the fixed angle rotation matrix using the negative of the orientation angles, the rotation-translation-orientation solution is complete. For the perspective view, multiplication by the combined perspective-offset-scale matrix completes the transformation.

D. SUMMARY

It has been shown that translation and viewing-plane orientation can be accomplished in one matrix transformation, namely,

$$H = \begin{bmatrix} \lambda_1 & \lambda_2 & \lambda_3 & 0 \\ \mu_1 & \mu_2 & \mu_3 & 0 \\ \nu_1 & \nu_2 & \nu_3 & 0 \\ X_T & Y_T & Z_T & 1 \end{bmatrix} \begin{bmatrix} \lambda_1 & \lambda_2 & \lambda_3 & 0 \\ \mu_1 & \mu_2 & \mu_3 & 0 \\ \nu_1 & \nu_2 & \nu_3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & -Y_0/F & -Z_0/F & -S/F \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & Y_0 & Z_0 & S \end{bmatrix}$$

Object Axes	Viewing Plane	Transformation
Rotation (Euler angles) Translation Viewing Plane Translation	Orientation (fixed axes rotation)	Perspective Offset Scale

In implementing this, the first rotation matrix and both translation matrices are combined. The result is post-multiplied by the viewing-plane orientation matrix. The result is modified to reflect the scale and perspective transformations and if offsets are present, that modification is also made. The result remains a 4 x 4 matrix that can be used to transform all the points to be displayed and that need be calculated only once for each view desired.

IV. DATA STRUCTURE

The type of data structure to be used in any given graphics program is greatly dependent on the purpose and nature of the program. The complexity can vary greatly due to the variety of applications. Three-dimensional data structures will necessarily be more complicated than two-dimensional data structures. A graphics program that will be modified on-line requires a higher degree of complexity than one which makes no provision for change.

A brief look into a few techniques adaptable to problems of increasing complexity will be described in this chapter. There are of course many ways of structuring data, and the following methods are merely an illustration of some of the characteristics and properties associated with data structures.

A. BASIC STRUCTURES

The two requirements in storing data are generally contradictory. Data should be rapidly available but at the same time it should be efficiently stored since memory is usually limited. The structures used in this section are not meant to be an exhaustive list of possibilities, but are rather illustrations of where one might begin in a search for a structure suitable to a specific problem.

1. Structuring for Points Only

The simplest of all data structures is that represented as an array of planar points. Consider the points in Figure 5 making up the letter 'A'. By using enough points, the 'A' may be made to appear to be composed of lines. Considerable memory is required in this type

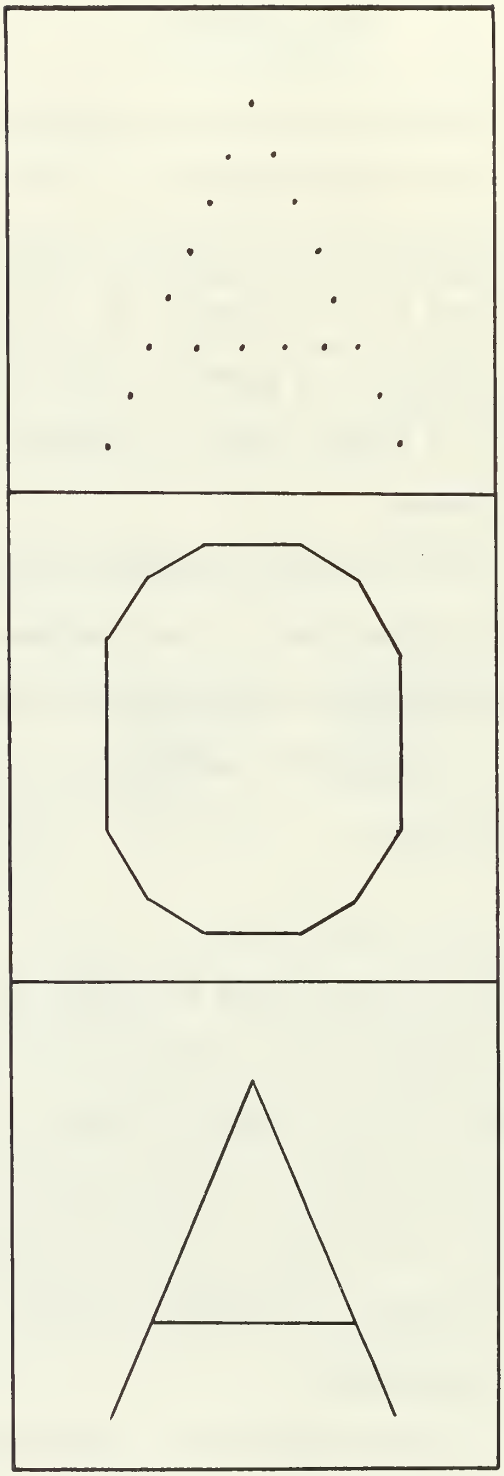


FIGURE 5

EXAMPLES OF DIFFERING DATA STRUCTURE REQUIREMENTS of storage, due to the large number of points needed as the picture becomes more complicated. Each point requires two coordinates.

The points could be stored as fixed point numbers unless the loss in resolution is unacceptable, such as in the case of magnification of details. A 24-bit computer word can yield a resolution of one part in 2^{23}, assuming signed numbers and the full computer word are being used. Since present display devices give at best one part in 2^{12} and many yield only one part in 2^{10}, packing these point coordinates into one word is possible for further memory saving. These techniques can be used to advantage in assembly language programming, but are of limited use in FORTRAN IV.

2. Structuring for Connected Lines

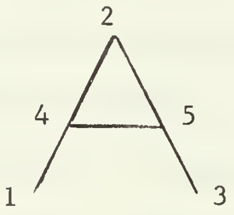
Increasing the complexity by one step, the figure '0' of Figure 5 is drawn. This figure is composed of connected straight-line segments. The points can be stored again as in the previous case, but now a line is drawn to each succeeding point. The line is continuous, although broken, and therefore no other special information is necessary.

3. Structuring for Line Segments

Going next to the 'A' made up of line segments, a higher-order storage problem is encountered. Now the figure is not made up of just one connected line, but rather a connected broken line and a line segment, or more simply just three line segments. Now it is necessary to note the line segments as well as the points making up the line segments. If just the points are stored, there is no way of knowing how these points should be connected for display. If just the lines are stored, the end points are undefined. If the lines are stored listing the endpoints, there is unnecessary repetition of the same points. It is necessary therefore to store a block containing the points and another block that defines the lines that exist between designated points. Figure 6 shows a structure for the 'A', with the points numbered for clarification. Although the line end points can be referred to by their value or location, storage location was used in this example since this gives more flexibility and points are not entered redundantly. The line list of Figure 6 contains the identification number of the points used to draw the 'A'. The points are defined in the point list. To display the figure, the line list is traversed, moving from line 1 to line 3. Line 1 is drawn between points 1 and 2 by referring to the point coordinates in the point list.

POINT LIST

1	x_1	y_1
2	x_2	y_2
3	x_3	y_3
4	x_4	y_4
5	x_5	y_5



LINE LIST

1	1	2
2	2	3
3	4	5

FIGURE 6

DATA STRUCTURE FOR THE LINE 'A'

4. Structuring for Simple Three-Dimensional Figures

In the two-dimensional case, storage in the line list of the actual points is probably preferred, making for a simpler structure. On the other hand, as the third dimension is introduced, it is preferable to define the lines by the location of storage of the point coordinates. This is so simply because the three-dimensional conversion is done on the points. To keep from converting the same point several times because it is stored in different locations, it is desirable to make all references to the point by location. For example, in Figure 7 each end point is a member of three lines. In the three-dimensional case it is not desirable to fit the three coordinates into one word (by packing) because the resolution would be severely limited--in the case of a 24-bit computer word the resolution for the packed case would be 1 out of 2^7, or little better than 1 out of 100. Any calculations that are to be made on the coordinates would degrade the accuracy further due to round-off error. Additionally, the coordinates would have to be unpacked for each operation or series of operations and then returned to the packed mode. Therefore three-dimensional data must be stored using more than one memory location. The display points after conversion could be packed for display and stored as before since only two coordinates are displayed in the pseudo 3-D presentation.

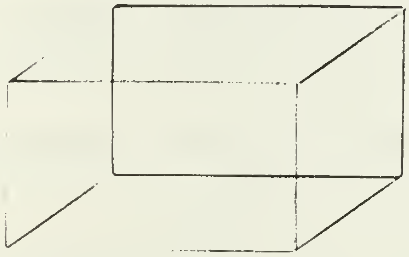


FIGURE 7

WIRE-FRAME RECTANGULAR BLOCK

The converted points could of course be stored in the place of the original points, but the original data is then lost. To draw the wire model of the block, the line list can be followed, with each line drawn as in the previous case.

5. Increased Complexity

Now consider the same rectangular block made up of six plane surfaces limited by their intersections. It now becomes necessary to identify the surfaces. The method chosen here is to describe each plane surface by a series of points, ordered so that the outside surface is to the left of the path described by the series of defining points. If the surface is not closed, this also must be noted. (For example, if the rectangular block were an open box, the object is not closed.)

Now there is a list for points, a list for lines, and a list for polygons defining limits of plane surfaces. Each line is a member of two surfaces and it is undesirable to draw the same line more than once. Some of the lines are hidden from view so that it becomes necessary to indicate which lines can be seen and which cannot. Additional information is therefore required. Examples of the proposed structure are shown in Figures 8-11. With the proposed structure, it is necessary for the display points to be stored by association with the points from which they were derived. Otherwise, reconstruction for display of the figure would be impossible.

The point data block of Figure 8 contains the uvw coordinates describing the object in three dimensions using the object's own coordinate system. Also contained in this block is the depth coordinate after transformation and the horizontal and vertical coordinates for display of the points. The number of lines to which the point belongs is stored in the first integer cell. Only three pointers are used in the data block, indicating to which lines the point belongs. The number of lines entering and exiting a point generally does not exceed three. There is, of course, the possibility of additional lines

<table>
<tr><td>Number of Points</td><td>Pointer #1</td></tr>
<tr><td>Pointer #2</td><td>Pointer #3</td></tr>
<tr><td colspan="2">u coordinate</td></tr>
<tr><td colspan="2">v coordinate</td></tr>
<tr><td colspan="2">w coordinate</td></tr>
<tr><td colspan="2">depth homogeneous coordinate</td></tr>
<tr><td colspan="2">horizontal display coordinate (y')</td></tr>
<tr><td colspan="2">vertical display coordinate (z')</td></tr>
</table>

FIGURE 8

POINT LIST DATA ARRAY

terminating at a point, and in this case another point block is used, as though the point were split.

The structure array block shown in Figure 9 varies with type of structure described. The structure shown is for a polygon defining the limits of a plane surface. The direction cosines of the inward-pointing normal are computed and stored. There are then eight pointers, each pointing to a line that defines the polygon. The type of structure (in this case a polygon) is coded by number in the second integer cell. The first integer cell contains the number of parts contained (the number of lines for a polygon) and thus is available as a pointer when entering data. If the number of parts exceeds the capacity, a second block is used. This method of storing data dictates that the next block is to be used for continuation (i.e. i + 1 is the continuation of i), and the indication that a continuation does exist is made by placing a number larger than the capacity in the first integer cell. In the block i + 1, a different type (code) number is assigned, indicating that this is a continuation of the previous block.

Figure 10 shows the block storage as it is allocated by physical location by the FORTRAN IV compiler. Since the first subscript varies first, the entire block, N by 8 in size, makes up the point list at the low end of the allocated storage. This is followed by an N by 2 block containing the line list. The structure list follows with another N by 8 block. Each division of Figure 10 represents N locations for real numbers. To split a location of a real number into two integer locations so that pointers can be inserted and read, the EQUIVALENCE statement is used. The integer array is dimensioned (2,N,18), where the length of the integer must be specified to be one-half the length

Structure Array (i)

<table>
<tr><td>Number of Parts</td><td>Type of Structure</td></tr>
<tr><td colspan="2">λ_N</td></tr>
<tr><td colspan="2">μ_N</td></tr>
<tr><td colspan="2">ν_N</td></tr>
<tr><td>Pointer #1</td><td>Pointer #2</td></tr>
<tr><td>Pointer #3</td><td>Pointer #4</td></tr>
<tr><td>Pointer #5</td><td>Pointer #6</td></tr>
<tr><td>Pointer #7</td><td>Pointer #8</td></tr>
</table>

Line Array (i)

Starting Point #	End Point #

FIGURE 9

DATA STORAGE IN STRUCTURE AND LINE ARRAYS

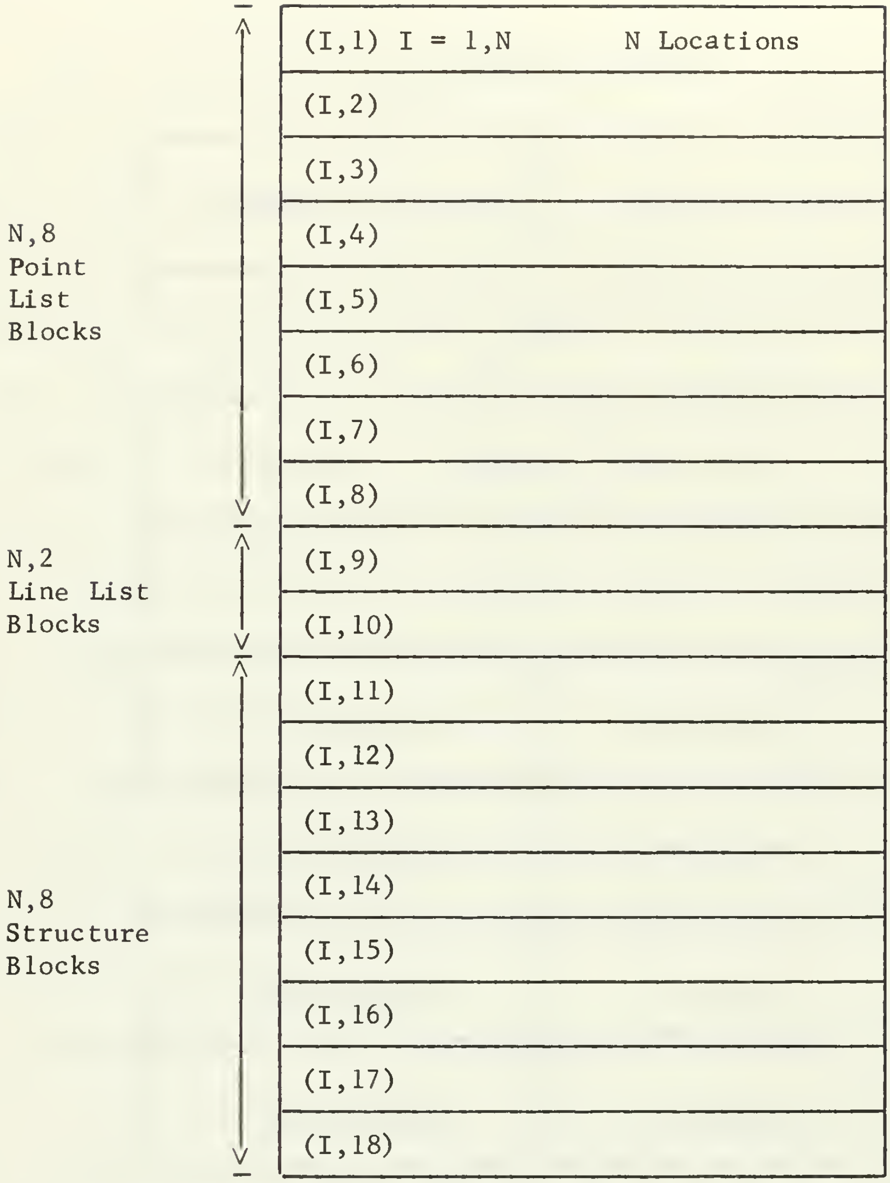


FIGURE 10

ARRANGEMENTS OF STORAGE LOCATIONS

of the real number. The EQUIVALENCE statement sets (1,1,1) of the integer array equal to (1,1) of the real-number array. Thus each real-number location can also be referenced by a pair of integer locations. This can be depicted as two integer cells packed into one real-number cell.

ARRAY (I,J)

INTEGER ARRAY (1,I,J)	INTEGER ARRAY (2,I,J)

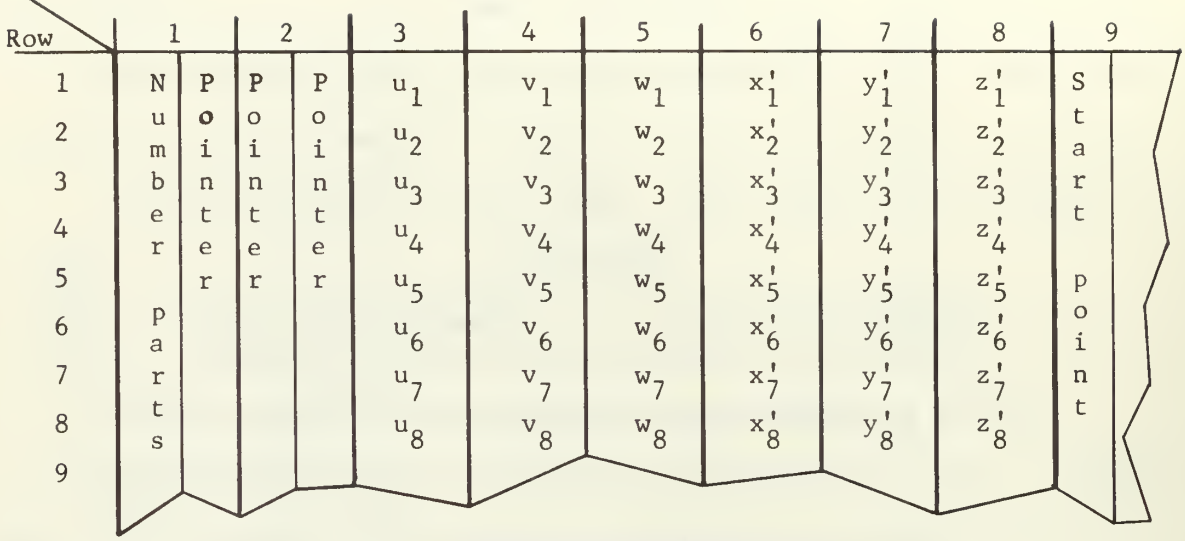
Another convenient way to visualize the structure is shown in Figure 11. Here the structure is depicted as a matrix dimensioned N by 18. The first eight columns make up the point list. The next two columns make up the line list and the last eight columns make up the structure list. Some columns are divided as explained above to utilize space more effectively. The row number N is the "name" of the point, line, or structure in that row.

No provision has been made here for linking various structures together. It is assumed that in displaying the image, the structure blocks will be traversed in order until all the blocks utilized have been traversed. If data is to be added or subtracted on-line from the basic structure, additional pointers would be necessary, pointing both forward and backward so that elements can be added or removed. In addition, it becomes necessary to employ an identification scheme so that if lines are members of arrays no longer used, they can be checked to see if they are used in some other array, and if not, they can be eliminated.

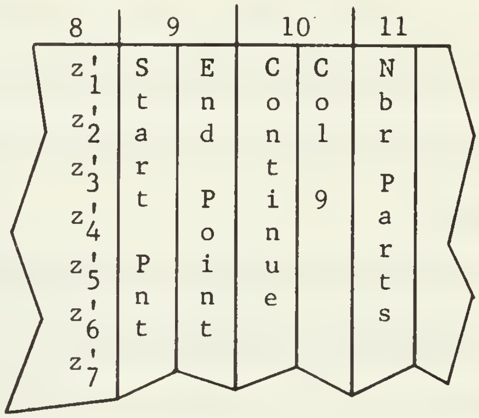
B. STRUCTURE FOR HIDDEN-LINE REMOVAL

If the object which is to be described is allowed to have concave dihedrals (where intersecting plane surfaces form inclusive angles

POINT LIST BLOCKS


Column
Row
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
Number parts
Pointer
Pointer
Pointer
u_1 u_2 u_3 u_4 u_5 u_6 u_7 u_8
v_1 v_2 v_3 v_4 v_5 v_6 v_7 v_8
w_1 w_2 w_3 w_4 w_5 w_6 w_7 w_8
x'_1 x'_2 x'_3 x'_4 x'_5 x'_6 x'_7 x'_8
y'_1 y'_2 y'_3 y'_4 y'_5 y'_6 y'_7 y'_8
z'_1 z'_2 z'_3 z'_4 z'_5 z'_6 z'_7 z'_8
Start point

LINE LIST BLOCKS



STRUCTURE LIST BLOCKS

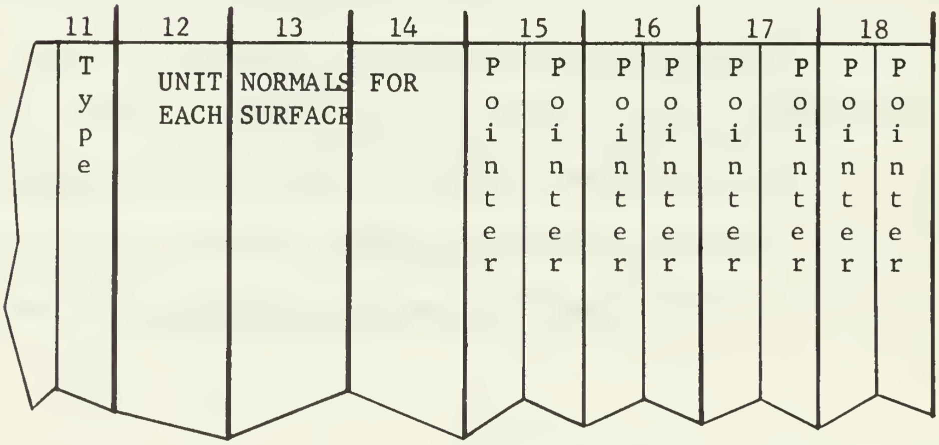


FIGURE 11

MATRIX INTERPRETATION OF DATA STRUCTURE

external of the object of less than 180^{o}), additional information is needed. This section will discuss how the description of the object and developed information is stored in association with nodes (points), faces (surfaces), and edges (lines). The utilization and development of the stored information will be discussed in chapter VI.

The implementation of the three-dimensional structure for the display of objects with concave and convex dihedrals and the elimination of their hidden lines is somewhat similar to that described in the last section. Some data was reorganized, some was dropped, and some was added. The result is a modified structure which contains the elements required by Galimberti and Montanari [Ref. 18] in their paper on the solution to the hidden-line problem.

It was desired that the dimensioning requirements for a user be kept to a minimum. Therefore one large array is used which contains several line lists, a node (point) list, and a polygon list. Two integer cells are again equivalent to one real-number cell.

1. Node (Point) Information

The basic structure must always rest on the individual nodes. The remaining stored data must provide the information necessary to connect the nodes for the display of the desired picture.

An N x 6 array was used for the node uvw coordinates in real space (referenced to the object axes) and for the depth, horizontal, and vertical coordinates in display space. The first six locations (columns) were used so that the conversion subroutine could be programmed to transform the first three locations and store the result in the second three locations.

Also associated with the node is information concerning the set of faces hiding that node and whether or not a node is an endpoint of a concave dihedral. The convex/concave determination is made and the result is entered in the first integer cell of the seventh position in the array.

In the thirteenth position of the array, the number of on-view faces hiding the associated node is placed in the first integer cell and a pointer to the first such face for each node is placed in the second integer cell. The pointer indicates a starting position in a single-subscript list that contains the blocking faces for all nodes, entered as they are encountered. Figure 12 summarizes the information for the nodes.

2. Faces (Surface or Polygon) Information

The eighth position in the array is divided into two integer locations. The first integer cell contains the number of points (nodes) defining the face and the second contains a pointer to the starting point in a list of nodes. If a face is hidden by its own volume, the first integer cell is marked with a negative sign.

Also associated with the faces are the inward-pointing normals. These are computed and stored in the fourteenth to sixteenth position of the array. The normals are used to compute concave surfaces, and when that computation is completed, the normals are no longer needed. The storage space for the normals then becomes a temporary storage area during the examination of segments for the elimination of hidden lines.

The list of defining nodes is a single-subscript array containing the nodes (by 'name') in the order in which they are entered. Figure 13 summarizes the information for faces.

Column	1	2	3	4	5	6	7		13	
Node 1	μ_1	ν_1	w_1	x_1'	y_1'	z_1'	0-convex 1-concave	not used	No. of blocking faces	pntr to list
2	μ_2	ν_2	w_2	x_2'	y_2'	z_2'				
3	μ_3									
4	μ_4									
N										

FIGURE 12

NODE INFORMATION

BASIC ARRAY

Row (Face)	Column 8	
1	± Nº of Points	Pointer
2		
3	+ indicates an on-view face	
4		
5	- indicates a face hidden by its own volume	

FNORML

Row (Face)	Column 1	2	3
1	x_{n1}	y_{n1}	z_{n1}
2	x_{n2}	y_{n2}	z_{n2}
3			
4	components of unit normal to each associated face		
5			

The FNORML array can also be interpreted to be columns 14-16 of the basic array.

LSTFAC

Point Number	
1	Number (name) of each point entered
2	
3	
4	
5	

LSTFAC is a single-subscript array.

FIGURE 13

FACES INFORMATION

3. Edge (Line or Segment) Information

The remaining locations from nine through twelve are used for information concerning lines. Locations nine and ten form one list, with ten following sequentially after all room is taken in nine. Location eleven and twelve form a similar list. Both lists are made up of two integer cells at each real-number location. The first list contains the starting and end point numbers (names) of the edge. The second list contains the two faces that join to make the edge.

The edge information is ordered in these lists. The lower-numbered (named) node is always stored in the first cell and the face with which the orientation corresponds is entered in the first cell of its list.

A negative sign is used to mark the first node in the ninth location to indicate that the segment is hidden (and that the segment is 'erased'). The second integer cell is marked to indicate whether the segment has been examined. The first integer location in the eleventh position is marked for concave lines.

Unused space in both line lists is used for temporary storage when determining intersections and when determining the nature of a point (i.e. the set of faces that hide that point). Figure 14 summarizes the structure for lines. An application example may be found in the chapter on hidden-line removal.

C. APPLICATION DEPENDENCE

It should be remembered at this point that the structures herein discussed are FORTRAN IV oriented. Further packing of information and pointers would be possible by resorting to the assembly language for a

Row (line number)	Column					
	9		10	11		12
1	start nodes	end nodes	continuation of 9	face of line in col. 9	face with reversed orientation	continuation of 11
2						
3	- hidden	- examined				
4	+ on view	+ not examined				
5				- concave		
6				+ convex		
7						
8						
9						

FIGURE 14

LINES INFORMATION

given computer. Reduction to machine-oriented programming can introduce efficiencies in many areas at the expense of machine independence.

Many structures have been proposed. The usual discussion of structures however is machine-oriented or in a higher-level language than FORTRAN. Most discussion in data structuring deals with the two-dimensional case. Little has been done in FORTRAN because of its inefficiencies and inherent disadvantages when compared to other languages. A number of references dealing with structures may be found in the Bibliography. The requirements for efficient storage and rapid access are contradictory and trade-offs must be made, dictated in part by the application and the computer to be used.

V. CONTOUR LINES

Contour lines find use in many fields. As techniques are developed for working with contour lines, a greater variety of applications will appear. In engineering applications, contour lines could be used when working with field problems such as

lines of equipotential,
lines of constant pressure,
lines of equal acoustic intensity,
and lines of equal radiated power intensity.

This discussion of contour lines will discuss a few of the fundamental concepts that could be applied in a graphics program. The subject is worthy of study by itself and thus a complete treatment is beyond the scope of this presentation. The Bibliography contains a number of references that deal with contour lines.

Specifically, the aspect of contour lines of immediate interest is the method of handling them so that they may be displayed in perspective. A function of two variables usually describes a surface. Many problems could be solved on sight if the right picture could be obtained of that surface. Two steps are required. First the contour lines must be obtained and second the desired picture must be constructed.

A. DEFINITIONS

One definition of a contour as given in the dictionary is "an outline". The contour lines will be used to construct an outline of an object or surface. Contour lines are normally defined as the lines of intersection between a given surface and a family of parallel surfaces (usually horizontal planes). S. P. Morse [Ref. 34] points out that this definition is ambiguous if the given surface has a portion parallel to the parallel surfaces. For this reason, Morse defines positive and negative contour lines. Since contour lines in topography are most easily visualized, the following definition will be given in that reference. The positive contour line is a line connecting points all of the same elevation such that points adjacent to one side of the line are at a higher elevation and points adjacent to the other side are at the same or lower elevation. The negative contour line is a line connecting points all of the same elevation such that points adjacent to one side are at a lower elevation and points adjacent to the other side are at the same or higher elevation. The usual contour line on a map is generally a union of these two. Obviously there must be yet another kind of contour line to satisfy the cases of maxima and minima (peaks and valleys). These are called degenerate contour lines as opposed to the normal contour lines. Specifically they are called maximum and minimum contour lines. The positive direction of a normal contour line is such that the points of higher elevation are on the left-hand side and points of lower elevation are on the right-hand side as the contour line is traversed in its positive direction.

B. ENCODING CONTOUR LINES

For storage in a computer, a contour line can be represented by an encoded string. A rectangular grid system proposed by H. Freeman [Ref. 14, 15] is a simple method of relating each point from previous points. A rectangular grid overlay is placed on a contour map. The curved contour line is then quantized to discrete values by assigning the closest node as the discrete contour point. To move from a reference point R in Figure 15 to the next point, one of eight allowed directions must be taken. Each direction is assigned an octal integer. Each odd number represents a movement of 1.414 units at an odd multiple of 45 degrees. Each even number and zero represents a movement of 1.0 units at an even multiple of 45 degrees. The agreement between quantized values and the actual value on the contour line is a function of the grid spacing. See Figure 16 for an example of an encoded string.

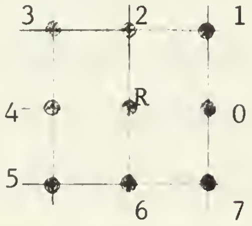


FIGURE 15

FREEMAN'S RECTANGULAR GRID

The grid size can be adjusted to yield the resolution required. B. W. Boehm [Ref. 5] suggests that a variable-size reference grid might be more appropriate. Thus, when little change in a surface occurs, a relatively large grid could be used. When the surface changes rapidly the grid size can be reduced so that details can be seen.

A contour encoded with a small grid size can be converted to a larger grid size by "integrating" a number of encoded digits. The number of digits thus integrated determines the new grid size. This would be helpful in seeing the large scale view but would maintain the detail for the close-up views (i.e. by retaining the original grid in computer storage).

Besides the encoded string, each contour line must have associated with it the two-dimensional coordinates of the first point and the elevation of the contour line.

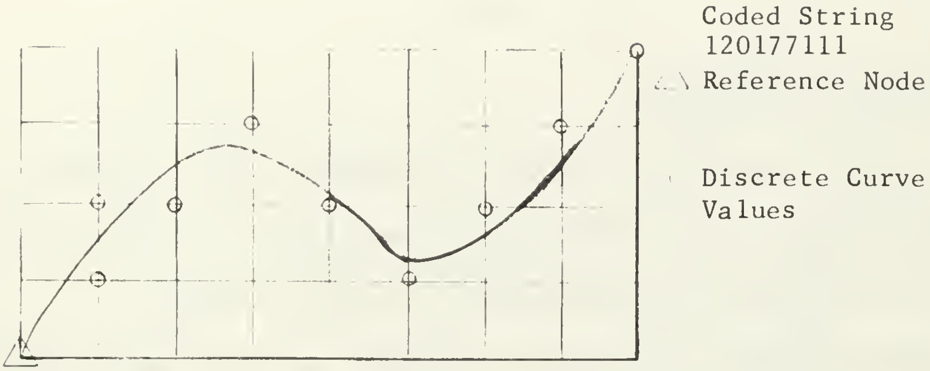


FIGURE 16

ENCODED STRING

C. OBTAINING CONTOUR LINES

Topographical contour lines are available on contour maps. In the engineering field, the data must be gathered (as of course it must originally have been done for topographical contour lines) and the contours constructed. Data can be gathered with regular, ordered samples or by random samples. Often no choice is possible. In the case of a surface which is a function of two variables, the data can be spaced at regular intervals. The intervals can also be varied for additional detail where the surface varies rapidly. Data obtained from weather observations may take on the characteristics of random sampling. In any case the data should be collected with the realization that it will be used to draw contour lines. If the horizontal space for which the contour lines are to be drawn is divided into a grid of size appropriate for the resolution desired, and the values of the horizontal planes are quantized at appropriate levels, the contours should become evident when the data is taken (or estimated, based on observations available). A contour line passing through a node must cross at or near one of the adjacent nodes unless the contour line does not extend beyond the immediate region of the first node.

D. DISPLAY OF CONTOURS

Once the contour lines have been obtained, selected points can be converted for perspective display of the lines. The obvious points to use are those points corresponding to the grid nodes.

The "cardboard cutout" type of display would be the simplest to obtain. In this type of display, the outline of the object is obtained and displayed as if the object were a flat surface perpendicular to the line of sight. To accomplish this, it is necessary to sense the left and right limits of visibility (from the observer's viewpoint) of each contour line and then connect all the left extremities and all the right extremities within a family of curves.

1. Determination of Limits of Visibility

The determination of the left and right visibility-limiting points can be done by maximizing and minimizing the tangent function of the angle between the y axis of the viewing plane and a given contour curve. Figure 17 shows six possible cases.

Only those contour points that show negative depth coordinates need be examined. All positive depth coordinates are behind the viewing plane. In both quadrants in front of the viewing plane, a minimum tangent function occurs at the left edge and a maximum tangent function occurs at the right edge. These are arithmetic maxima and minima. The only difference in the two quadrants is that in the left-hand quadrant

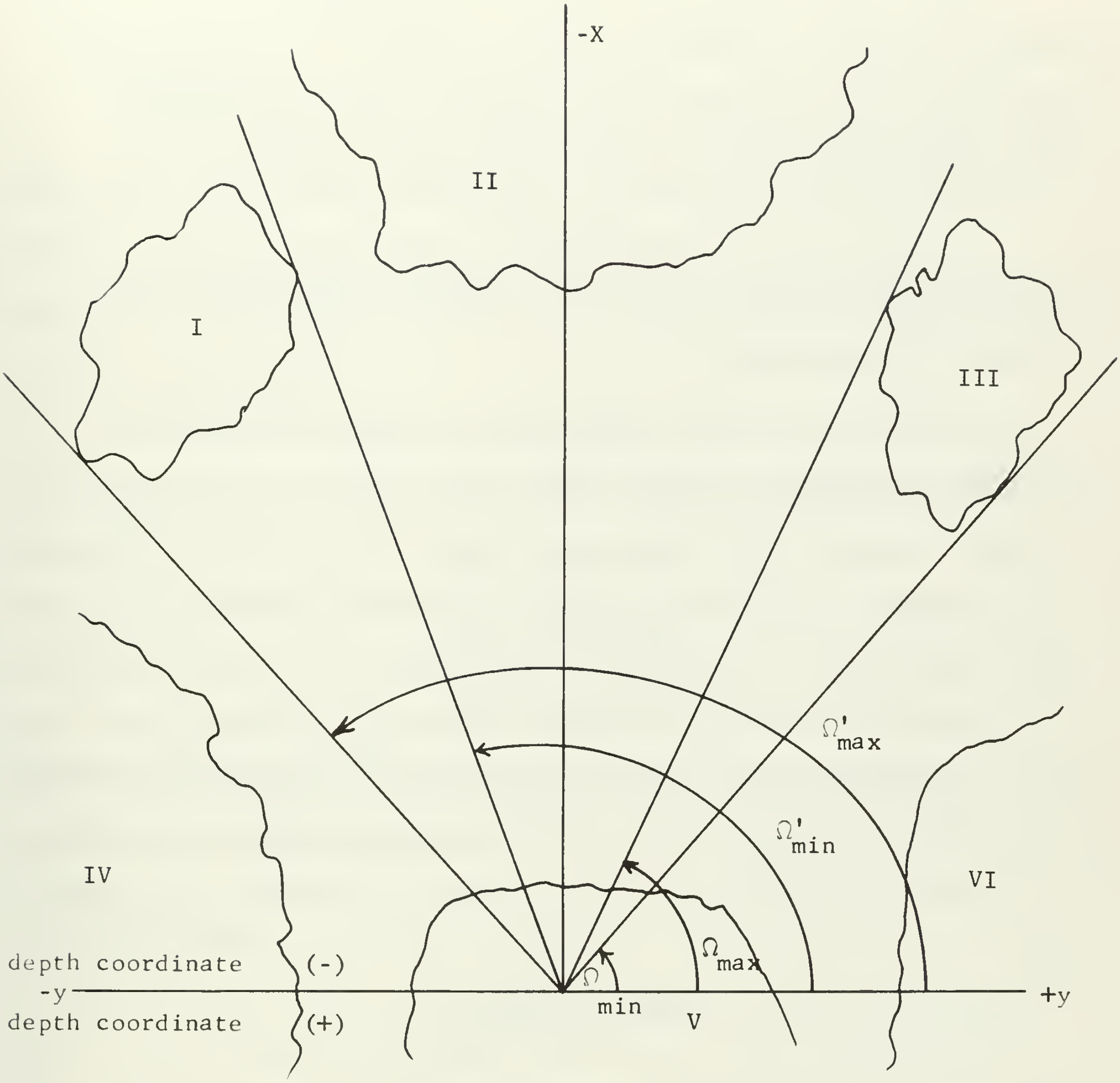


FIGURE 17

CONTOUR LINES EXAMINED FOR LEFT AND RIGHT EXTREMES the sign of the tangent function is positive whereas in the right-hand quadrant the sign of the tangent function is negative.

The actual angle need not be found. The magnitude of the tanget function is easily evaluated by using the origin of the viewing plane and the point on the contour line evaluated in the same coordinate system. The coordinates of a point located on a contour line are

$$u_c \; v_c \; w_c$$

and the combined movement of the object axes and viewing-plane axes results in the quantities

$$X_T \; Y_T \; Z_T$$

and previously defined in chapter III, i.e.

$$X_T = X_{OB} - X_{VP}$$

$$Y_T = Y_{OB} - Y_{VP}$$

$$Z_T = Z_{OB} - Z_{VP}.$$

Only two dimensions need be used at this point, with the angle Ω defined as the angle that a half-plane coincident with the positive y axis and the yz plane sweeps about the z axis to reach the contour line extremity. Thus,

$$\tan \Omega = \frac{u_c + X_T}{v_c + Y_T}$$

and the point where this is a maximum arithmetic value is a right extremity. The point where this is a minimum arithmetic value is a left extremity. Note that $(u_c + X_T)$ will be negative for points in front of the viewing plane.

A treatment of the general case of the elimination of hidden lines for two dimensions is detailed in an article by H. Freeman and

P. P. Loutrel [Ref. 17]. Such an elaborate procedure was not desired in this case. This is a method of improving detail, but of course costly in computing time and storage.

2. Variations

A more complete picture can be obtained by also displaying those parts of the contour lines that lie in front of the limits of visibility (and are not hidden by other surfaces). The price of additional detail is again additional computation time and more memory storage requirements.

The method of hidden-line elimination discussed in the following chapter can be applied to the outlines derived. The hidden-line elimination is applicable to objects drawn with plane surfaces. Therefore it is necessary to assume the cardboard-cutout type of display for at least that part of the program that eliminates the hidden lines. The three points defining a plane must be chosen very carefully however, since different combinations of left and right limit points will generally define different planes.

E. FURTHER STUDY

As three-dimensional capabilities are expanded, the use of contour lines in a wide variety of problems will begin to look more attractive. Morse points out that there are many problems which might be effectively solved by using contour maps to represent the data if techniques for processing contour maps were more fully developed [Ref. 34].

VI. HIDDEN-LINE REMOVAL

One of the major problems in three-dimensional display work has been the removal of hidden lines. A simple two-dimensional procedure was discussed in the section on contour lines. Generally, methods that have been used have been limited in application. Some solutions apply only to convex polyhedra. In order to process a large vocabulary of solids, point-visibility schemes employing "brute force" methods were used. The cost of such methods was a very large computation time.

The method applied in this study uses ideas and concepts from reports by A. Appel [Ref. 2] and R. Galimberti and U. Montanari [Ref. 18]. Appropriate modifications were made to shorten computation time wherever possible. This method is applicable to convex and concave plane-faced objects. The number of objects is not limited (except by total storage availability), but all objects must be described in a single coordinate system.

A. THE PREVIOUS REPORTS

The two previous reports used as a basis for this treatment of the hidden-line problem defined some concepts for the first time. In his report, Appel gave a general solution to the hidden-line problem for a large vocabulary of objects. Appel's method reduced the computation time from previous methods by an order of magnitude. Galimberti and Montanari presented the solution of the hidden line problem for either convex or concave plane-faced objects. Some of Appel's concepts were used by Galimberti and Montanari. The computation time was further reduced but at the expense of some generality.

1. Quantitative Invisibility

Appel's analysis of the picture parts was done in three-dimensions, and can be extended to curved surfaces as well as plane surfaces. The concept of quantitative invisibility was introduced by Appel to determine which lines or parts of a line were to be drawn. Quantitative invisibility is concerned only with a count of the number of surfaces hiding a point. For those portions of a curve where the count is zero, the curve is drawn. When the count is other than zero, the curve is being hidden by a surface or surfaces between the curve and the observer and the curve is not drawn. Appel introduced the concept of implied vorticity and suggested a method for determining whether a point lies interior or exterior to a bounded surface. Both of these concepts are employed in this solution to the hidden-line problem and will be explained in a later section.

2. Invisibility by Sets

Galimberti and Montanari extended Appel's method by making use of the projected lines and points and extending the concept of quantitative invisibility to become the concept of qualitative invisibility. Instead of the number of surfaces hiding a point, Galimberti and Montanari used the idea of an identified set of surfaces hiding that point. Their application was limited to convex and concave plane-surfaced objects. Any disadvantage introduced by the requirement that the object be described in this form was offset by the relative speed with which the calculation was made. Galimberti and Montanari outlined their algorithm very well. This study follows their algorithm in a general way, with many modifications.

B. DEFINITIONS

The many concepts that will be discussed are at times difficult to visualize. Even greater confusion arises when the same word is applied to mean two different things. A clear distinction must be made between names in two dimensions and names in three dimensions. The terms defined here will be used throughout the remainder of the chapter.

An object is assumed to be made up of faces which are bounded portions of plane surfaces. The intersecting faces form a concave dihedral if the included angle (measured external of the object) between the two intersecting faces is less than 180°. Faces are bounded by edges, where an edge is the intersection of two planes. The end points of an edge are called nodes. An edge is convex or concave, corresponding to the dihedral of the faces which define the edge. A node is concave if it belongs to at least one concave edge. Projection on a picture plane yields polygons for faces, segments for edges, and vertices for nodes. A point is a general term and is used in both two and three dimensions. Both edges and segments are composed of a series of points. Nodes and vertices are special points.

Two segments cross at an intersection. A segment that intersects a second segment is said to be winning if the first corresponding edge crosses in front of the second corresponding edge. The second segment is said to be losing. Thus the intersection is a winning intersection for the first segment and a losing intersection for the second segment.

The nature of a point is the set of faces in three dimensions which block that point from view. To find the nature of a vertex, the corresponding node must be checked for any intervening faces. The set of intervening faces is the nature of the vertex (and the node). When the nature equals the null set the point is in view.

C. SPECIAL TESTS

Various special tests were developed to be used with this program. Many of these tests have uses beyond the immediate applications in this solution to the hidden-line problem. In each case the tests have been reduced to a minimum of programming in an attempt to minimize time and computer memory.

1. Implied Vorticity

If it is desired to determine whether two points are on the same or different sides of a line, a test for implied vorticity can be made. Refer to Figure 18. The sense of rotation about A in going from P_1 to P_2 is counter-clockwise, but the sense of rotation about B is clockwise.

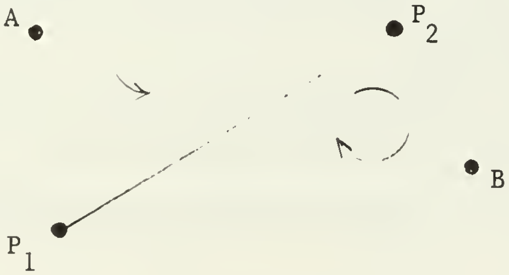


FIGURE 18

IMPLIED VORTICITY

The area of a triangle bounded by three points can be found by

$$A = \pm \frac{a}{2} \begin{vmatrix} Y_0 & Z_0 & 1 \\ Y_i & Z_i & 1 \\ Y_{i+1} & Y_{i+1} & 1 \end{vmatrix}$$

$$A = \pm \frac{b}{2} \begin{vmatrix} Z_0 & X_0 & 1 \\ Z_i & X_i & 1 \\ Z_{i+1} & X_{i+1} & 1 \end{vmatrix}$$

$$A = \pm \frac{c}{2} \begin{vmatrix} X_0 & Y_0 & 1 \\ X_i & Y_i & 1 \\ X_{i+1} & Y_{i+1} & 1 \end{vmatrix}$$

where a, b, and c are the direction cosines of a line perpendicular to the plane of the triangle determined by the points P_0, P_i, and P_{i+1}. The sign is a function of the direction of rotation about P_0. The actual sign is not of consequence for the applied purpose, but a change in sign in testing two points indicates a change from one side of the line to the other. The area is of no concern for this application and therefore the direction cosine is not needed. The equation as used for testing is used in the two dimensional case and reduces to the form

$$AH = X_0Y_1 + X_2Y_0 + X_1Y_2 - X_2Y_1 - X_1Y_0 - X_0Y_2$$

where AH is an indicator and

$(X_0\ Y_0)$ are the coordinates of the point under test, and

$\left.\begin{matrix}(X_1\ Y_1)\\(X_2\ Y_2)\end{matrix}\right\}$ are the coordinates of the end points of the line.

This can be modified to

$$AH = X_0Y_1 + X_2Y_0 - X_1Y_0 - X_0Y_2 + C$$

where

$$C = X_1Y_2 - X_2Y_1$$

This was implemented as a function statement for rapid evaluation.

When applying the test, obviously two points must be evaluated or the test is meaningless. The point being tested is always $(X_0\ Y_0)$

and is being tested in respect to the line with end points (X_1 Y_1) and (X_2 Y_2).

To test for an intersection of two lines, the test for implied vorticity must be applied twice. First each of the end points of one segment must be tested with respect to the end points of a second segment. If no change in sign occurs, implied vorticity is the same for both points and there can be no intersection. If there is a change in sign, each of the end points of the second segment must be tested in respect to the end points of the first line. Again, no change in sign indicates no intersection is possible. A change in sign in both tests indicates that there is an intersection within the segment lengths.

In the example of Figure 19, it is desired to test lines L_2, L_3, and L_4 for intersection with line L_1. Point C is tested with respect to line L_1 (points A and B). Point D is then tested with respect to line L_1. The signs of the two results are compared and in this case the signs are the same (either both positive or both negative). This indicates that no intersection can occur between these two lines. The actual polarity of the sign is determined by the direction in which line L_1 is traversed between points A and B.

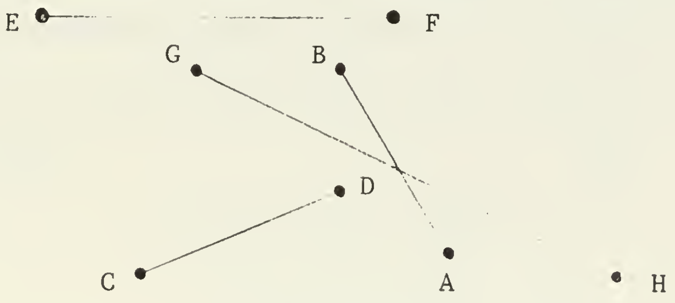


FIGURE 19

TESTING A LINE FOR INTERSECTION

Next, points E and F are each checked with respect to line L_1. This time a change in sign occurs since the implied vorticity is counter-clockwise for E and clockwise for F when the direction in line L_1 is from A to B. Now the possibility of an intersection exists, but when the test is applied to points A and B in respect to line L_3 it is seen that no intersection is possible.

Last, points G and H are tested with respect to line L_1 and it is again seen that the possibility of an intersection occurs. Testing A and B in respect to line L_4 completes the test and an intersection is indicated since the sign changed in both applications of the test.

Several special cases arise when one or both points are collinear with the line being tested. Figure 20 and the following examples explain the special cases that can develop.

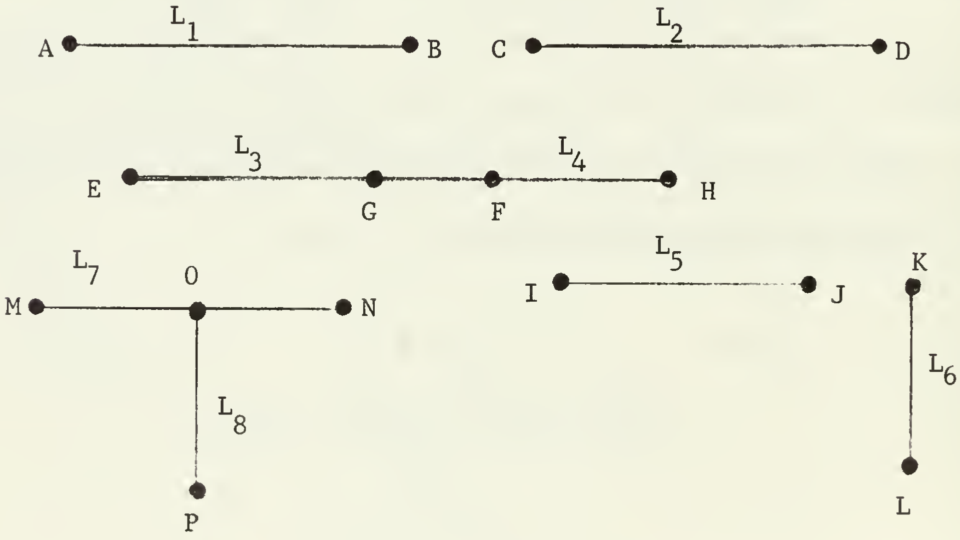


FIGURE 20

INTERSECTION SPECIAL CASES

In the first example, segments L_1 and L_2 belong to the same line, but the segments do not meet. In this case, the application of the vorticity test to either end point of one segment in respect to the other segment results in a zero (since the area of the triangle is zero). The same condition exists for L_3 and L_4, except that now there is an intersection. In the case of lines L_3 and L_4, two intersections are assumed, one at G and one at F (or at the end points of the common segment). If G and F coincide, only one intersection is assumed. In both of these examples the implied vorticity test fails to provide the desired information and a special test was developed.

When the result of the implied vorticity is sensed to be zero (or close to zero), a separate test is used. If the lines are not vertical, the x components of the vertices are used in the test.

Let

$$R = \text{minimum} \left[\text{maximum } (X_A, X_B), \text{ maximum } (X_C, X_D) \right]$$

$$L = \text{maximum} \left[\text{minimum } (X_A, X_B), \text{ minimum } (X_C, X_D) \right].$$

Then R - L will result in

1) a negative number if there is no intersection

2) zero if the intersection is at one common end point

3) a positive number if there is a common segment

Applying the test in the case of the line L_1 and the line L_2, it can be seen that

$$R = \text{minimum} \left[X_B, X_D \right] = X_B$$

$$L = \text{maximum} \left[X_A, X_C \right] = X_C$$

and

$$(R - L) = X_B - X_C = \text{a negative number, hence no intersection.}$$

For lines L_3 and L_4,

$$R = \min \left[\max (X_E, X_F), \max (X_G, X_H) \right]$$

$$L = \max \left[\min (X_E, X_F), \min (X_G, X_H) \right]$$

and

$(R - L) = X_F - X_G$ = a positive number, hence the segments must overlap.

If $X_F = X_G$, the lines intersect at a common end point and the result of R - L is zero. If the lines are vertical, the same test is applied with the Y coordinate in place of the X coordinate.

The remaining two special cases occur when only one of the end points is collinear with the other segment. Two different situations are shown in Figure 20. Lines L_5 and L_6 show no intersection and lines L_7 and L_8 show an intersection. Of course if points I and J are tested with respect to line L_6, it can be seen immediately that there is no intersection. But if points K and L were tested with respect to line L_5 first, an intersection may or may not be indicated, depending on the sign of the very small number resulting from testing K in respect to line L_5. Although this small number should be zero, round-off errors in conversion would probably cause it to be other than zero (although extremely small). Each result of the vorticity test is therefore checked for the small magnitude that would result from one end point being collinear. In this program only losing intersections are noted since only the losing intersections cause changes in qualitative invisibility.

2. Exterior - Interior Determination

The principle of this test was also suggested by Appel. If a line is drawn from the point being tested to a point exterior of the boundary, the number of crossings must be an odd number if the point is interior to the boundary or an even number if the point is exterior to the boundary.

Applying this test to the three-dimensional problem, the projection of a face onto the picture plane yields a closed boundary (a polygon). If some point (usually a node) is projected onto the picture plane also, and a line is drawn from that point to a point that is known to be exterior to the polygon, the number of crossings of the polygon will reveal whether the projected point is inside or outside the polygon. If the projected point is inside the polygon, the point is either in front of or in back of the face corresponding to the polygon. A depth test to determine which is closer to the observer will resolve the ambiguity. If the projected point is exterior to the polygon, the corresponding face cannot be hiding that point.

Figure 21 contains a polygon $P_1P_2P_3P_4P_5P_6P_7$ with a hole in it. The hole is bounded by $P_9P_{10}P_{11}P_{12}$. Point A lies inside the bounded area. If a line is drawn from A to any point outside of the polygon, an odd number of crossings of the boundary will occur (either 1 or 3). For point B, the number of crossings will be even (or zero). In this example if the known exterior point is beyond the limits of the polygon P_{1-7} (i.e. not in the hole), the number of crossings will be two. From point C the number of crossings will be zero, two or four, depending on the direction in which the line is drawn.

To reach a point exterior to the polygon, a half-line segment is assumed to be drawn vertically from the projected point. All segments are then checked for intersection with this vertical half-line. The faces defining the edges that correspond to the crossed lines are marked. When all segments have been checked, those faces which have been marked an even number of times cannot be hiding the point. Those faces marked an odd number of times are further checked for depth to determine which faces block that point.

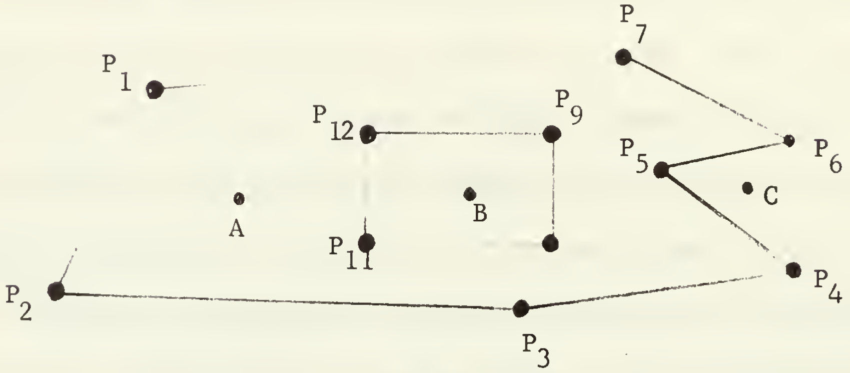


FIGURE 21

INTERIOR-EXTERIOR DETERMINATION

3. Intersection of a Plane and a Line (PIERC)

A subroutine was developed to provide the coordinates of the intersection of a plane specified by three points and a line specified by two points. This test is necessary to determine where a line connecting the observer and a point under test passes through a surface. If the intersection is further away than the point, the point is in front of the surface.

The subprogram was developed from the basic equations of a line and a plane. The equation for a plane specified by three points is

$$\begin{vmatrix} x - x_1 & y - y_1 & z - z_1 \\ x_2 - x_1 & y_2 - y_1 & z_2 - z_1 \\ x_3 - x_1 & y_3 - y_1 & z_3 - z_1 \end{vmatrix} = 0$$

The parametric form of the line equation is

$$x' = x'_1 + ta$$

$$y' = y'_1 + tb$$

$$z' = z'_1 + bc$$

where

$$a = x'_2 - x'_1$$

$$b = y'_2 - y'_1$$

$$c = z'_2 - z'_1$$

and t is the parmeter and the primes denote the coordinates of the points defining the line.

Expanding the equation for the plane and arranging terms

$$(x - x_1)\left[(y_2 - y_1)(z_3 - z_1) - (z_2 - z_1)(y_3 - y_1)\right]$$
$$+ (y - y_1)\left[(z_2 - z_1)(x_3 - x_1) - (x_2 - x_1)(z_3 - z_1)\right]$$
$$+ (z - z_1)\left[(x_2 - x_1)(y_3 - y_1) - (y_2 - y_1)(x_3 - x_1)\right] = 0.$$

Let

$$A = (y_2 - y_1)(z_3 - z_1) - (z_2 - z_1)(y_3 - y_1)$$

$$B = (z_2 - z_1)(x_3 - x_1) - (x_2 - x_1)(z_3 - z_1)$$

$$C = (x_2 - x_1)(y_3 - y_1) - (y_2 - y_1)(x_3 - x_1)$$

$$D = - Ax_1 - By_1 - Cz_1$$

Then

$$Ax + By + Cz + D = 0,$$

where A, B, and C are the direction numbers of a normal to the plane. To find the intersection of the plane and the line, substitute the expressions for x, y, and z defining the line into the equation for the plane.

$$A(x_1' + ta) + B(y_1' + tb) + C(z_1' + tc) + D = 0.$$

Solve for the parameter t.

$$t = \frac{Ax_1' + By_1' + Cz_1' + D}{-\ (Aa + Bb + Cc)}$$

The coordinates of the intersection then are

$$x = x_1' + ta$$

$$y = y_1' + tb$$

$$z = z_1' + tc$$

This test is implemented in the subroutine PIERC (pierce). A, B, C, and D are found as above using the three points defining the plane. The direction numbers of the line are then found. Combining these results allows the value of the parameter t to be found. The value of the parameter t is then used in the parametric line equation to find the intersection of the line and the plane. The entire process is coded into only eleven FORTRAN statements consisting of 29 addition/subtractions, 19 multiplications, and one division. The determination of the intersection of the plane and the line can thus be made very quickly.

D. ALGORITHMS

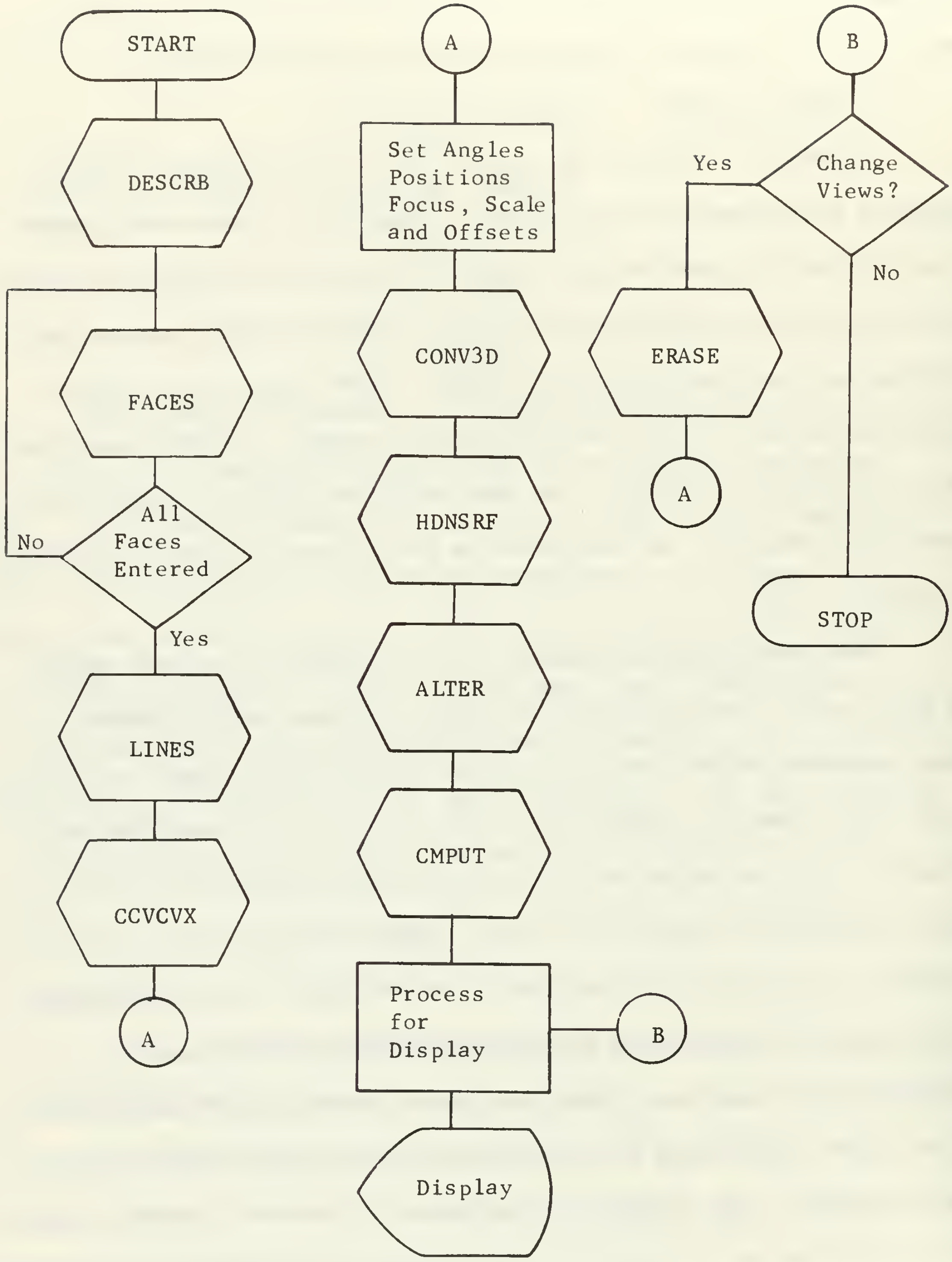
Although some portions of the program are not directly concerned with the hidden-line removal, there is an interacting influence between the hidden-line removal problem and the remainder of the program. The greatest influence is exerted through the storage of data and the associated requirements.

This section will explain the algorithms used in the various program parts. The first section will introduce a general program which could be used to generate a picture with hidden lines removed. Various subprograms are called. These subprograms will be further explained immediately following the introduction of the general program.

1. General Program

Flowchart 1 provides an overall view of the complete program. Dimension requirements are explained in Appendix C. Subroutine DESCRB is an initializing routine that resets all pointers and flags and clears certain portions of the array structure. Subroutine FACES enters that data for one face each time it is called. LINES connects the points which were entered to form the structure in storage. CCVCVX identifies all of the concave dihedrals. At this point the structure within the computer is complete. As many views as desired may be made by continuing on from A.

When developing the picture, it is necessary to set the angles and positions of the object and the observer before calling CONV3D. If these variables are not set the previous values will be used, or if this is the first view, all angles will be set to zero. All positions will be zero, except the object will be moved 40 units away from the focal plane, enabling the object to be seen. CONV3D converts the stored coordinates of each point to coordinates for horizontal and vertical display, and a depth coordinate which is associated with each point. Additionally the position of the focal plane center and two additional points in the focal plane are calculated in terms of the object's coordinate system. The position of the center of the focal plane is used when determining the coordinates of the intersection of the line of sight and a plane. This is used in subroutines VRTNTR and TSTNNP which are called by CMPUT. The three points in the focal plane are used to determine the coordinates of the intersection at the focal plane of a line



FLOWCHART 1

PROGRAM FOR GENERATING COMPUTER DRAWN PICTURE passing through the focal plane. This information is used in subroutine ALTER.

Some lines of a drawing will be hidden by the object's own volume. HDNSRF determines which surfaces are facing away from the observer and interprets this information to determine which lines are hidden by the object's own volume.

Subroutine ALTER will modify the line structure by terminating all lines passing through the focal plane and assigning new end points which lie in the focal plane to replace those end points which lie behind the observer. This is necessary due to the inverted image which is formed after passing through the focal point (similar to the image being inverted in a camera).

Each on-view line is then examined in subroutine CMPUT to determine which portions of the line are visible. This subroutine also calls the subroutines VRTNTR, TSTNNP, LNTRSC, WINDOW, and other routines of a more general nature. These subroutines will be discussed in the section explaining the algorithm for CMPUT.

At the completion of CMPUT, the picture has been "developed" and exists in the form of a list of starting point coordinates and end point coordinates for each visible line normalized to ± 1. The user must define the output to be used and must process the line list for any changes required from the normalized ± 1 and the output device. Two routines which are included in the demonstration program that accomplish this are DRAWVP and DRAWIT.

DRAWVP converts the line list to an output for the line printer. Up to ten symbols will be printed for each line. The picture will be distorted due to the constraint of printing only at discrete points, but offers the advantage of obtaining an immediate result which can be used to guide modifications to data describing the object. This routine loses its effect as the picture becomes more complicated. DRAWVP requires that subroutine VPLOT be furnished (see Appendix C for information on VPLOT).

DRAWIT is a subprogram developed to draw the output picture on the CALCOMP Plotter using the line list developed. DRAWIT utilizes the locally developed plotting package. (See Appendix C for further information on this subroutine and the plotting package.)

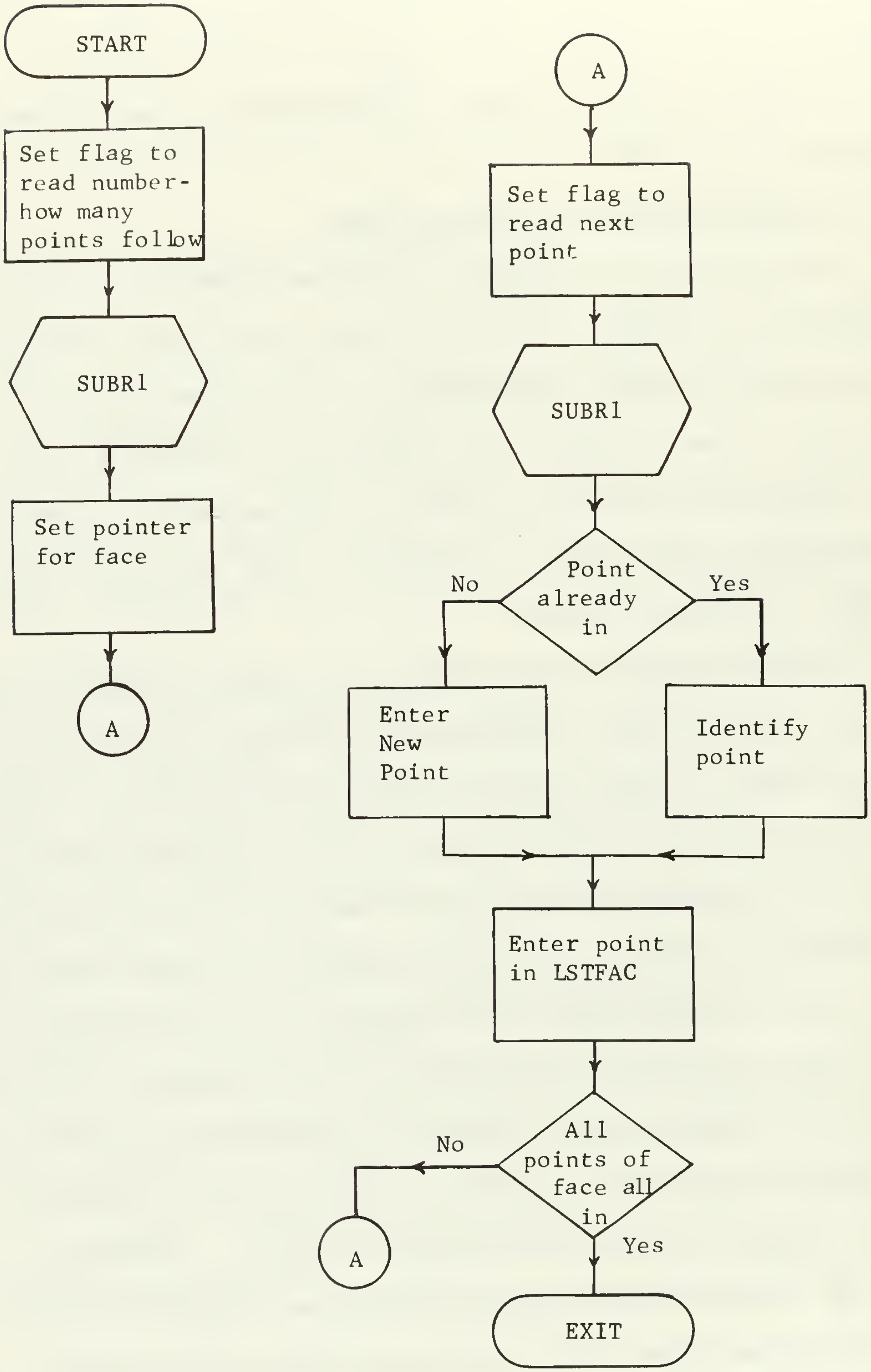
When the picture is completed, additional views may be desired. To obtain another view, subroutine ERASE is called to reset flags and pointers and to restore the line structure if it has been altered. The program is entered at the point of setting any new angles or positions or of setting new values of focal length (focus), offsets, or scale. If for example it was desired to magnify the picture by two with no other changes, scale would be reset to 1/4 (assuming it had been 1/2) and all other values left as already set. This means the normalized distance from the center to an edge of the picture is now 1/4 and this portion of the picture will be expanded to fill the "picture frame". This corresponds to enlarging a portion of the picture already made. For a telephoto effect, only the focal length would be changed (by increasing the value of the focus). This corresponds to using a telephoto lens in a camera. Although the results are similar, the two cases originate differently. In a camera the enlargement process is limited by the grain of the film but in the computer drawn picture the resolution is determined by the resolution used in the object description.

A main real array and an integer array occupy the same block of memory. Two integer words must equal one real word and this is assumed in the following discussion. EQUIVALENCE statements required are defined in Appendix C.

2. FACES (Subroutine for Entering Data)

The objects to be displayed are entered through the use of a subroutine called FACES. The objects to be entered must be described by bounded plane surfaces. The boundaries of the plane surfaces are defined by a number of points with straight lines connecting those points. The first point and last point of any such surface boundary are assumed to be connected. The first point must be a corner point for reasons explained in CCVCVX.

For each face, the first input is the number of points that are to follow. Then the three coordinates of each point are read until the specified number of points has been entered. As each point is read, it is compared to points already entered and if it is already present, an identifier n_i (a number) is assigned to that point and the coordinates of that point are stored in the (n_i,J) position of the main array (where J = 1, 2, 3). A list (LSTFAC) of the points in the order that they are entered is kept separately. The number of points entered for the face is stored in the main integer array at $(1,n_{if},8)$ where if is the number of the face being entered. A pointer to the start of the defining points in LSTFAC is stored in the array at $(2,n_{if},8)$. Flowchart 2 summarizes the algorithm for FACES. The subroutine must be called as many times as there are faces to be entered since only one face is entered by a single call. This subroutine may be called at any time that a face (or faces) is to be added to the existing structure, but



FLOWCHART 2

SUBROUTINE FACES

after the new face(s) has been entered, the subroutines LINES and CCVCVX must be called before proceeding with the development of the picture.

3. Connecting the Defining Points (LINES)

After the defining points have been entered, the points are connected by the subroutine LINES. Each point is to be connected by a straight line segment to the next point until all of the points for the face are used. The last point is then connected back to the first point.

In the line structure it is desired to store the end points so that the lowest numbered (named) point is always stored as the first point. The face for which this is the normal orientation of the line will be stored in the first integer of column 11 and the face for which this is the reverse orientation of the line will be stored in the second integer of column 11. The assumption here is that every line segment or edge is the intersection of two planes and will be entered once in each direction. If lines which lie entirely on one surface are to be entered they must be entered in both directions when that surface is processed. See Appendix C and the example input data and the program output following the Appendices.

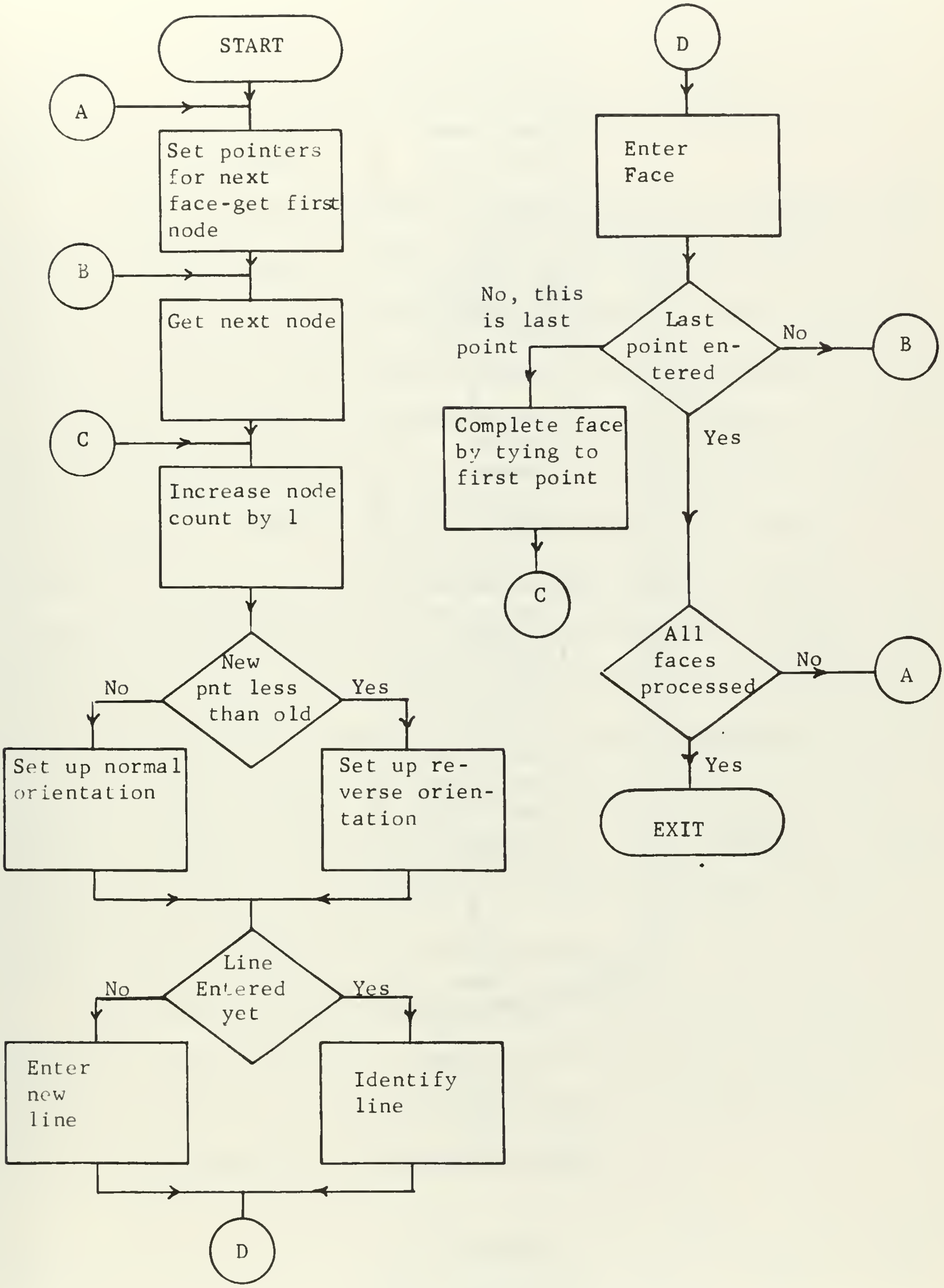
A line is only entered once. A search is made of the lines already entered. If the line is found, the face presently being drawn is entered in the proper integer in column 11 according to the orientation of the line. If the line is not found the line is entered by listing its end points in column 9. The present face is entered in the proper integer in column 11.

When the last point has been entered, the first point is entered as the new point and the final line for the face is processed. After the completion of this final line for the face being processed, the entire process is repeated for the next face. This continues until all of the input faces have had their sequential points joined by lines. See Flowchart 3.

4. Detection of Concave Dihedrals (CCVCVX)

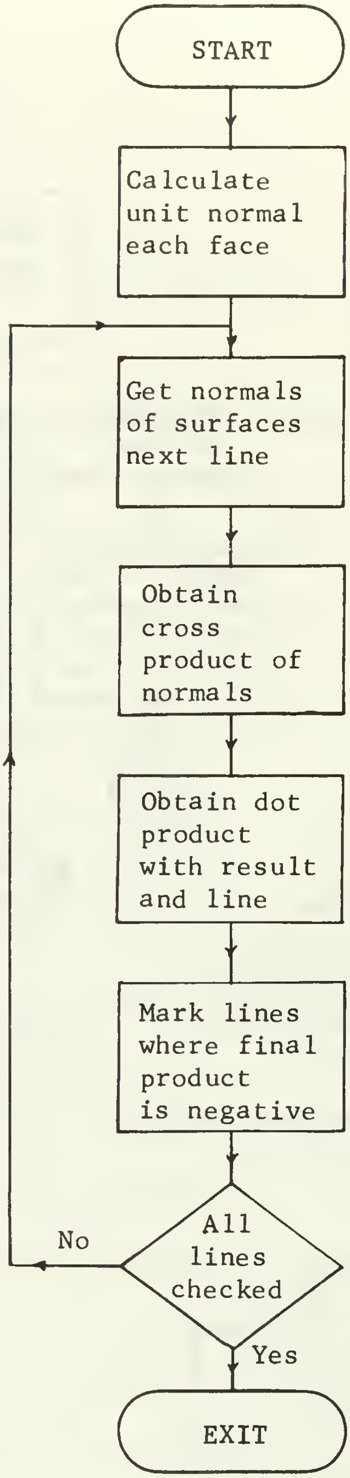
It is necessary in this treatment of the hidden-line problem to identify those edges which correspond to the intersection of two surfaces which form a concave dihedral. Subroutine CCVCVX identifies the concave dihedral and marks the appropriate line and its end points (Flowchart 4). A line which results from a concave dihedral is marked with a minus sign in the first integer in column 11. A node which is the end point of at least one concave line is marked with a 1 in the first integer of column 7. A zero in column 7 indicates that all lines of which that node is an end point result from the intersection of two surfaces forming a convex dihedral.

The first step in the detection of the concave dihedrals is the calculation of inward pointing normals for each face. The normal for a face is calculated by constructing a directed line from the first point entered to the last point entered (line 1) and a second directed line from the first point entered to the second point entered (line 2). The length of each of these lines is normalized. The cross product of the normalized lines [(line 1) x (line 2)] results in an inward pointing unit normal. The direction cosines of the normal are stored in array FNORML for use as required.



FLOWCHART 3

SUBROUTINE LINES


START
Calculate unit normal each face
Get normals of surfaces next line
Obtain cross product of normals
Obtain dot product with result and line
Mark lines where final product is negative
All lines checked
No
Yes
EXIT

FLOWCHART 4

SUBROUTINE CCVCVX

Each line is then examined by using the cross product of the normal of the first face in column 11 (normal orientation) and the normal of the second face in column 11 (reverse orientation). See Figure 22 as an example.

Assume line AB is defined such that A is the low numbered point and B is the high-numbered point. FACE 1 and FACE 2 are identified in the figure and correspond to the normal orientation of the line and the reverse orientation respectively. (Remember each surface must be entered by following the boundary such that the surface lies always to the left.) When the inward pointing normal of FACE 1 is crossed into the inward pointing normal of FACE 2, the result is a vector in the reverse direction of line AB. If the faces had formed a convex dihedral, the resultant vector would have been aligned with line AB. To detect the direction, the dot product of the resultant of the cross product and the line is calculated. This is the same as detecting the projection of the cross-product resultant on line AB. If the result of the cross product is negative, the faces intersect in a concave dihedral. If the result is positive, the intersection is a convex dihedral. A zero result indicates that the faces are coplanar.

The lines and end points are appropriately marked. When all lines have been checked, all concave dihedrals have been identified. This process need only be done when the structure is first entered.

5. Transformation to Perspective (CONV3D)

This is the first subroutine to be called which is view dependent. The rotation angles and positions of the object and the observer should be defined as desired before this subroutine is called.

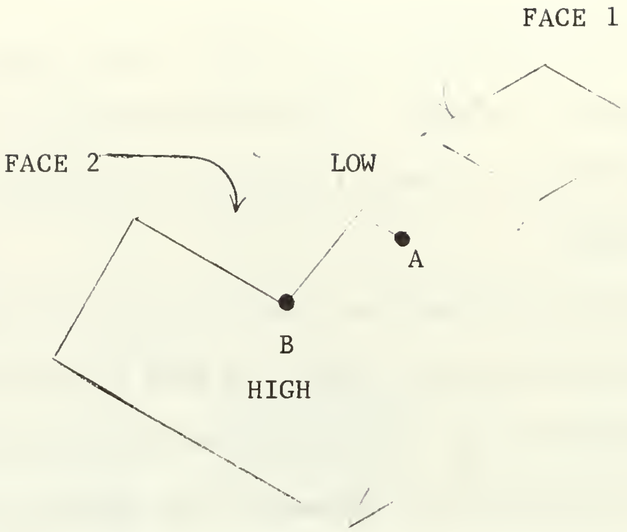
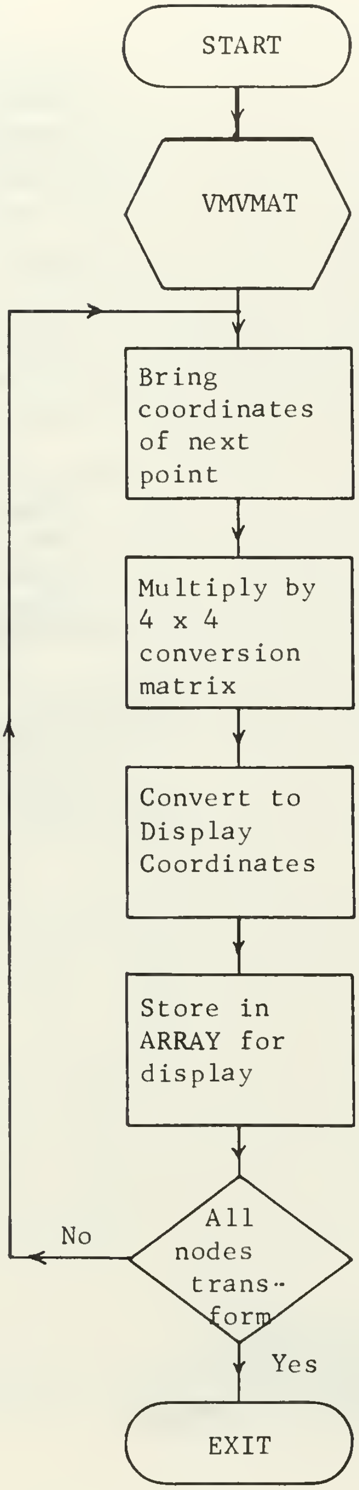


FIGURE 22

CONCAVE DIHEDRAL

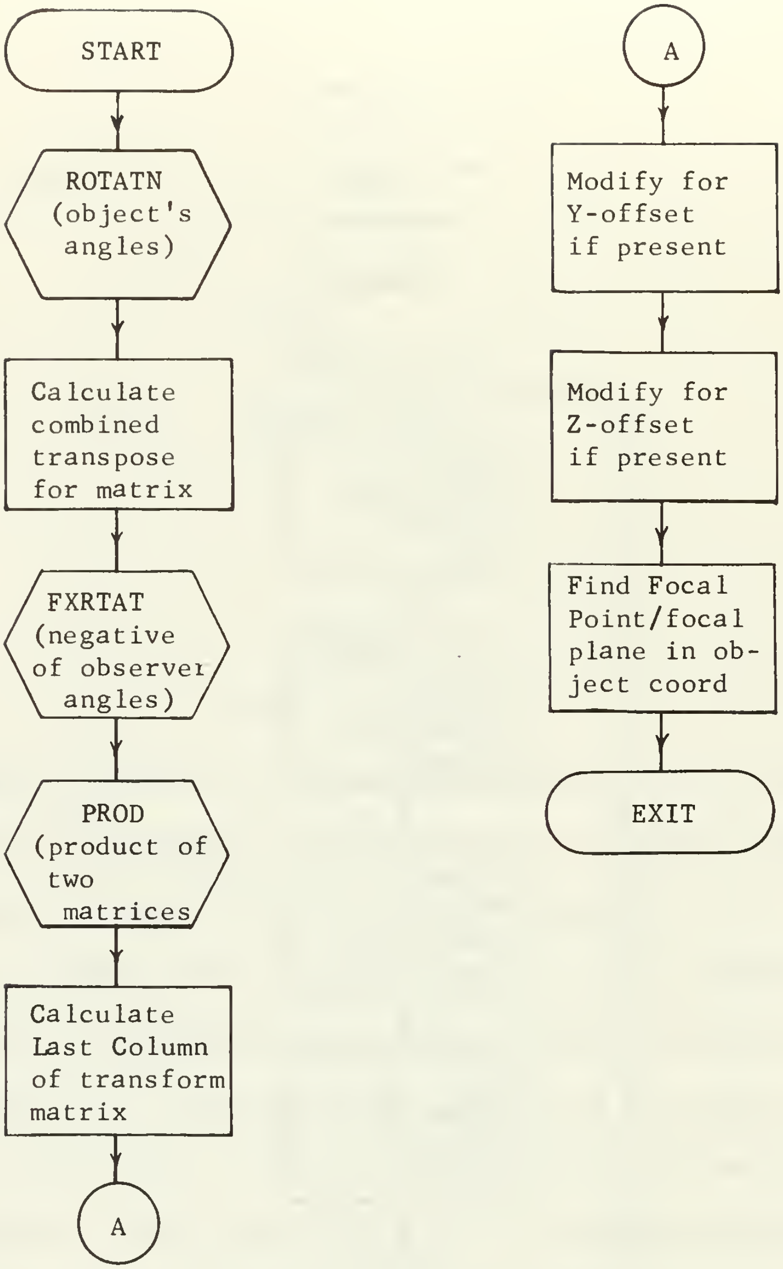
CONV3D (Flowchart 5) calls VMVMAT (Flowcharts 6, 7) as its first step. VMVMAT calculates the 4 x 4 homogeneous transformation matrix and the location of three points in the focal plane as well as the location of the focal point expressed in object axes coordinates. The computation of the homogeneous transformation matrix is discussed in chapters II and III. To calculate the location of the focal plane in terms of the object axes coordinates, Euler angle rotations were used with the observer's viewing angles and with the negative of the object's rotation angles. Once the matrix and the positions have been calculated, they remain available until the view is changed.

The three points used in the focal plane are the origin, the point y = 1 on the y axis, and the point z = 1 on the z axis. Translation of the observer less the translation of the object provides the resultant translation of the focal plane origin in terms of the object axes coordinates. By adding the proper direction cosines of the Euler


START
VMVMAT
Bring coordinates of next point
Multiply by 4 x 4 conversion matrix
Convert to Display Coordinates
Store in ARRAY for display
All nodes transform
No
Yes
EXIT

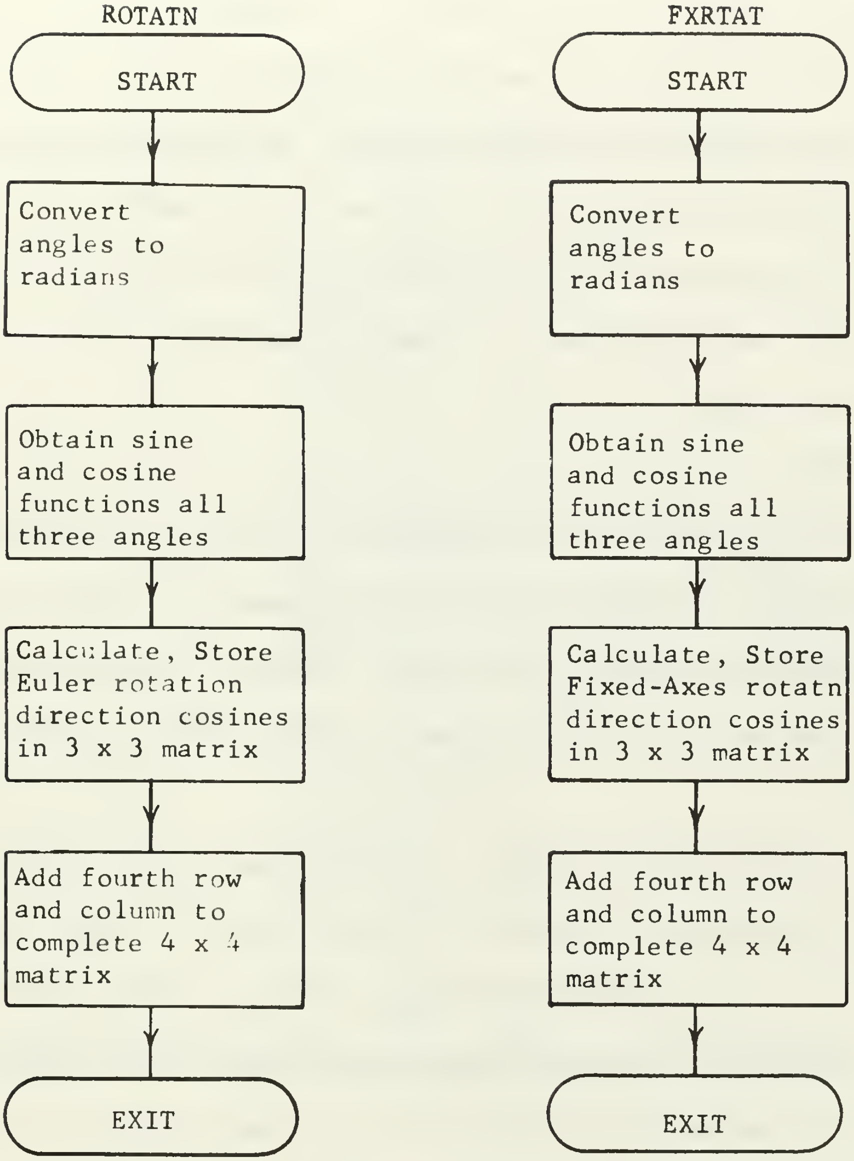
FLOWCHART 5

SUBROUTINE CONV3D


START
ROTATN (object's angles)
Calculate combined transpose for matrix
FXRTAT (negative of observer angles)
PROD (product of two matrices
Calculate Last Column of transform matrix
A
A
Modify for Y-offset if present
Modify for Z-offset if present
Find Focal Point/focal plane in object coord
EXIT

FLOWCHART 6

SUBROUTINE VMVMAT


ROTATN
START
Convert angles to radians
Obtain sine and cosine functions all three angles
Calculate, Store Euler rotation direction cosines in 3 x 3 matrix
Add fourth row and column to complete 4 x 4 matrix
EXIT
FXRTAT
START
Convert angles to radians
Obtain sine and cosine functions all three angles
Calculate, Store Fixed-Axes rotatn direction cosines in 3 x 3 matrix
Add fourth row and column to complete 4 x 4 matrix
EXIT

FLOWCHART 7

SUBROUTINE ROTATN AND SUBROUTINE FXRTAT

angle rotations of the observer, the three points in the focal plane are expressed in object's axes coordinates for translation of the object and observer and rotation of the observer. To compensate for object rotation, the Euler angle rotation cosines are calculated for the negative of the object angle rotation (using matrix form). Multiplying the three coordinates found previously for each point by the rotation matrix provides the coordinates of the three points in the focal plane in terms of the object axes coordinates. These are saved and used in a later part of the program.

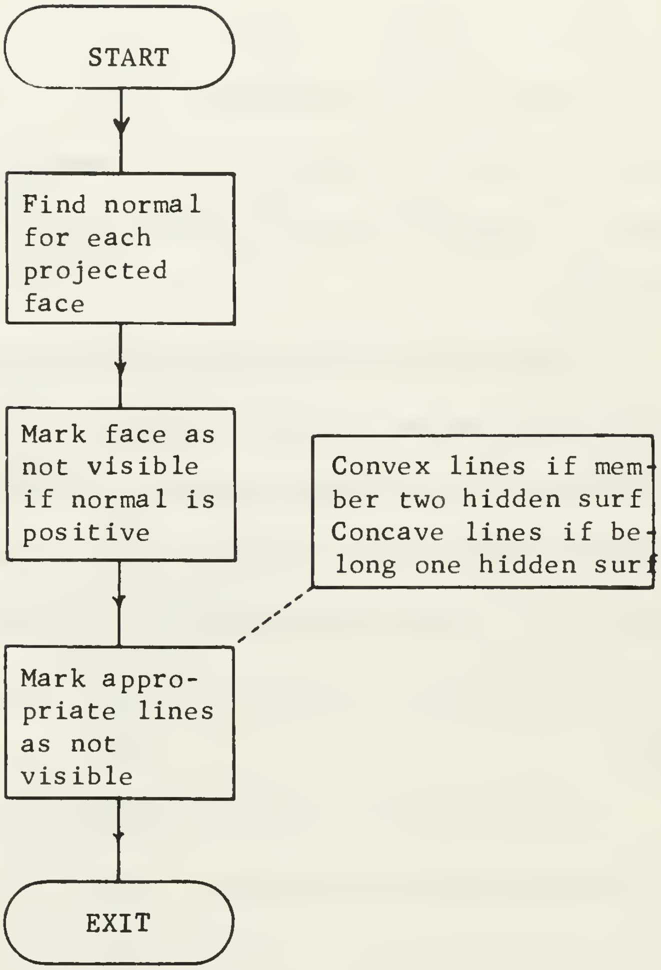
CONV3D proceeds with the conversion of all data points (or nodes). The theory of the conversion is explained in chapters II and III. The results of the conversion are placed in columns 4, 5, and 6 of the main ARRAY. Column 4 is a depth coordinate, column 5 is the horizontal picture-plane coordinate, and column 6 is the vertical picture-plane coordinate.

If the hidden lines were not to be removed, the picture could be drawn immediately by following the line list in column 9 and connecting the picture plane coordinates of the end points listed.

6. Suppressing Surfaces Hidden by the Object's Own Volume (HDNSRF)

In any given object constructed of plane surfaces, some of the surfaces will face away from the observer. Since these surfaces cannot be seen, it is desirable to eliminate them as soon as possible from further consideration and spend the computation time on the on view faces.

The test for a surface oriented away from the observer is made in the picture plane (Flowchart 8). Once again a line is constructed from the first point to the last point and from the first point to the


START
Find normal for each projected face
Mark face as not visible if normal is positive
Mark appropriate lines as not visible
Convex lines if member two hidden surf
Concave lines if belong one hidden surf
EXIT

FLOWCHART 8

SUBROUTINE HDNSRF

second point, where first, second, and last are in reference to the order in which the points for a face were originally entered. When the first line is crossed into the second line (cross product), the resultant vector will point either into the picture or out of the picture. Since the lines are coplanar, there are only two dimensions involved and the test is rapid. If the resultant vector is positive, the face is not on view and that face is marked with a negative sign in front of the first integer of column 8 (the integer that specifies the number of points in the face).

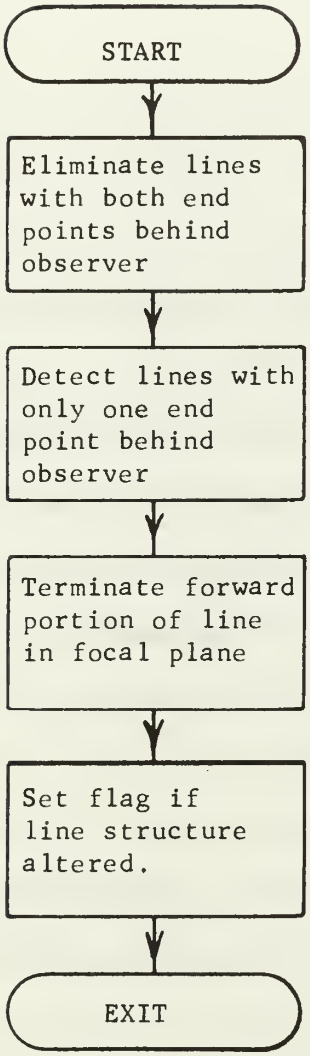
When all the faces are thus checked, the list of lines containing the intersecting planes is checked. If a line is the result of the intersection of two hidden surfaces, the line cannot be seen and is therefore marked with a negative sign in front of the first integer in column 9. If a line corresponds to a concave dihedral and at least one of the intersecting surfaces is hidden, the line is hidden and is so marked.

At this point, if a single object is being displayed and the object has no concave dihedrals, it is possible to draw the object with hidden lines removed by following the line list and connecting the end points of those lines not hidden.

7. Termination of the Picture in the Focal Plane (ALTER)

This subroutine need only be called when a portion of the picture lies behind the observer. In order to properly display the picture, the lines which pass through the focal plane must be terminated at the focal plane. Subroutine ALTER accomplishes this task (Flowchart 9).

Each line is examined by its end points to determine if either or both ends are behind the observer. The depth coordinate of the end


START
Eliminate lines with both end points behind observer
Detect lines with only one end point behind observer
Terminate forward portion of line in focal plane
Set flag if line structure altered.
EXIT

FLOWCHART 9

SUBROUTINE ALTER

point is negative if the point is in front of the observer (the observer looks into the negative x half-space). If both ends are in front of the observer, the line is not disturbed. If the line is hidden it is not considered. If both ends of the line are behind the observer, the line is marked as if hidden.

In the remaining two cases, one of the end points is in front and one is behind the observer. (Since the points are identifiable, the two permutations define two cases.) The three points in the focal plane that have had their coordinates converted to the object axes system are used in the PIERC subroutine to determine the coordinates of the intersection of the focal plane and the line. The coordinates of the intersection are added to the list of points as an additional point and the line structure is altered to reflect the change in end points for the line being examined. The coordinates of the intersection are converted with the homogeneous matrix to picture-plane coordinates and the result is stored as in the original data points.

When exiting this subroutine, a flag is set if any line structure has been altered. When ERASE is called, detection of this flag will cause subroutine LINES to be called to re-establish the original line structure.

8. Subroutines Called During Segment Examination

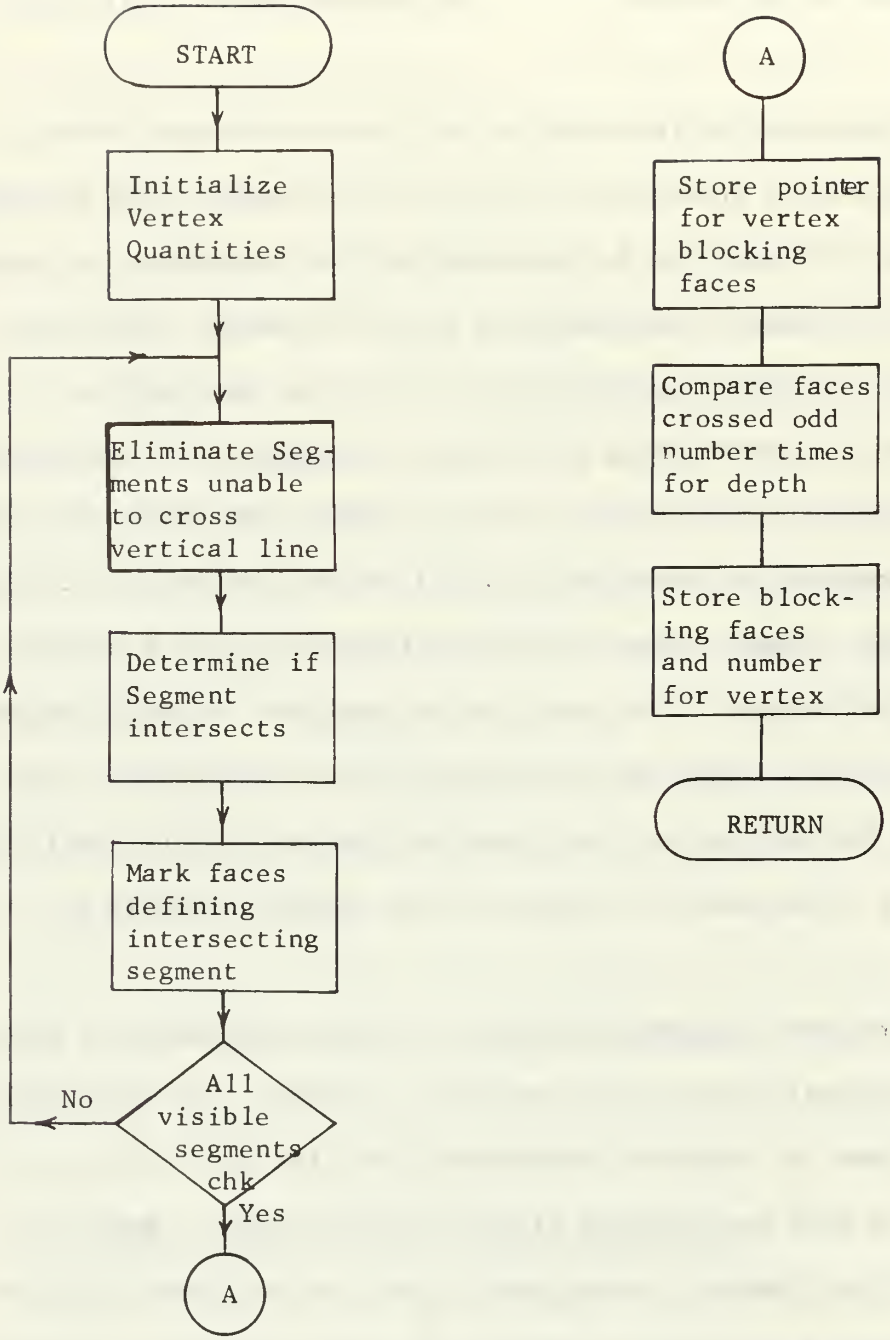
Segment examination is accomplished by subroutine CMPUT which in turn calls on certain other subroutines. Those subroutines which are called during segment examination will be discussed in this section.

a. Determination of the Nature of a Vertex (VRTNTR)

When examining a segment, it is necessary to know the nature (the set of faces obscuring the node) of the initial vertex (picture-plane point corresponding to the node). Subroutine VRTNTR is used to determine the nature of the vertex if it is not already known. (Flowchart 10 applies.)

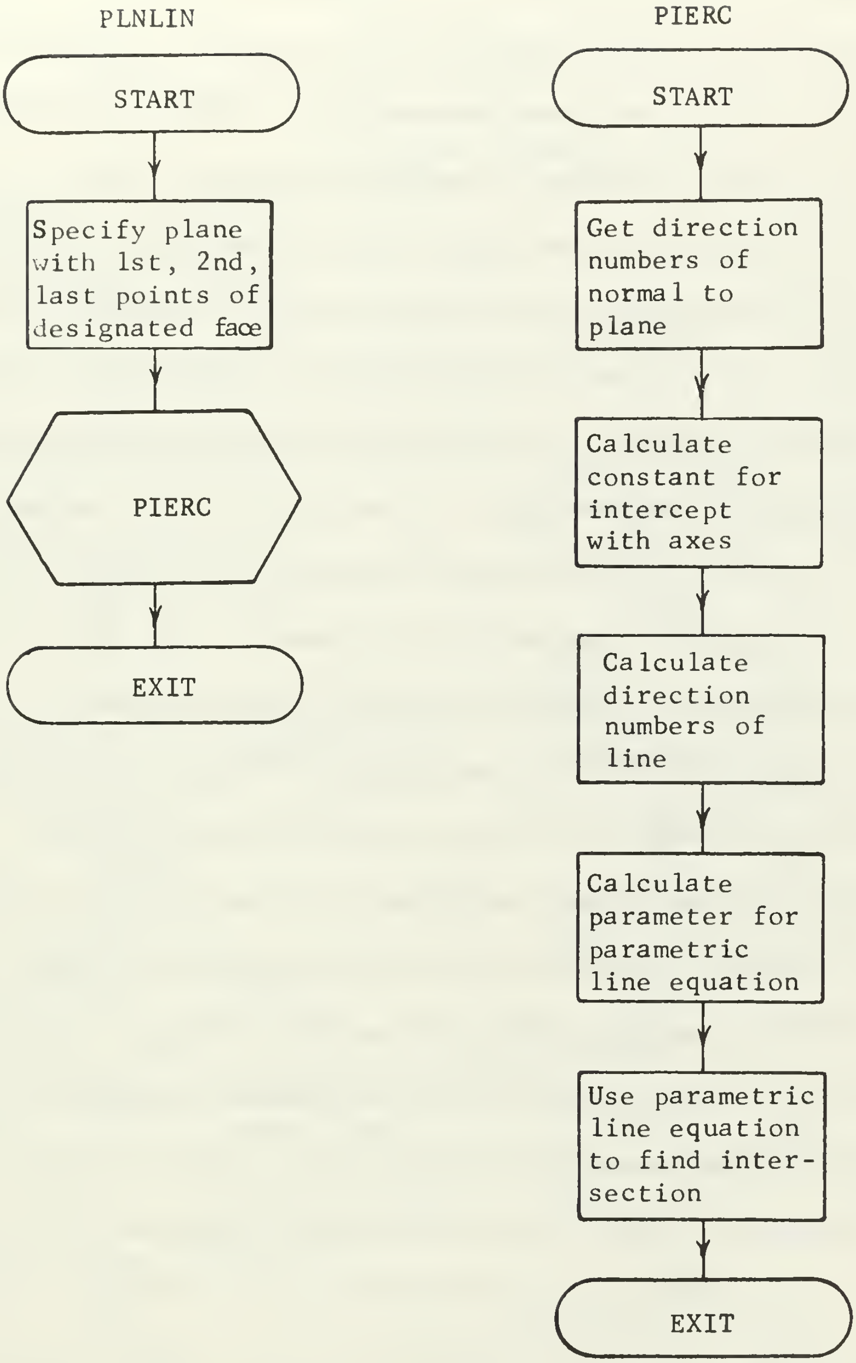
The basis for this test is the Exterior-Interior Determination discussed in an earlier section of this chapter. The half line is constructed vertically by allowing the vertical coordinate to exceed the vertical coordinate of any point on any line segment. All line segments that lie entirely to the left, entirely to the right, or entirely below the vertex being checked are eliminated from further consideration during the test of this vertex. Those lines which are hidden because the intersecting surfaces which define them are not on view are also eliminated. However those segments which are part of a concave dihedral and are members of at least one on view face are also checked even though the line itself may be hidden. This occurs because those boundaries of the on view face which are not concave dihedrals will be visible, and a wrong count will result if the concave dihedral is excluded.

For those segments which have not been eliminated, a test for intersection with the vertical half line is made. The faces which intersect to form the segments intersecting with the half line are marked once for each such segment if the face is on view. When all segments have been checked, those faces crossed an odd number of times are further checked to determine whether the face is behind or ahead of the node associated with the vertex. This is accomplished by calling PLNLIN which sets up the coordinates of three of the points determining the face and the coordinates of the points determining the line between the observer and the node being checked (Flowchart 11). The origin of


START
Initialize Vertex Quantities
Eliminate Segments unable to cross vertical line
Determine if Segment intersects
Mark faces defining intersecting segment
All visible segments chk
No
Yes
A
A
Store pointer for vertex blocking faces
Compare faces crossed odd number times for depth
Store blocking faces and number for vertex
RETURN

FLOWCHART 10

SUBROUTINE VRTNTR


PLNLIN
START
Specify plane with 1st, 2nd, last points of designated face
PIERC
EXIT
PIERC
START
Get direction numbers of normal to plane
Calculate constant for intercept with axes
Calculate direction numbers of line
Calculate parameter for parametric line equation
Use parametric line equation to find intersection
EXIT

FLOWCHART 11

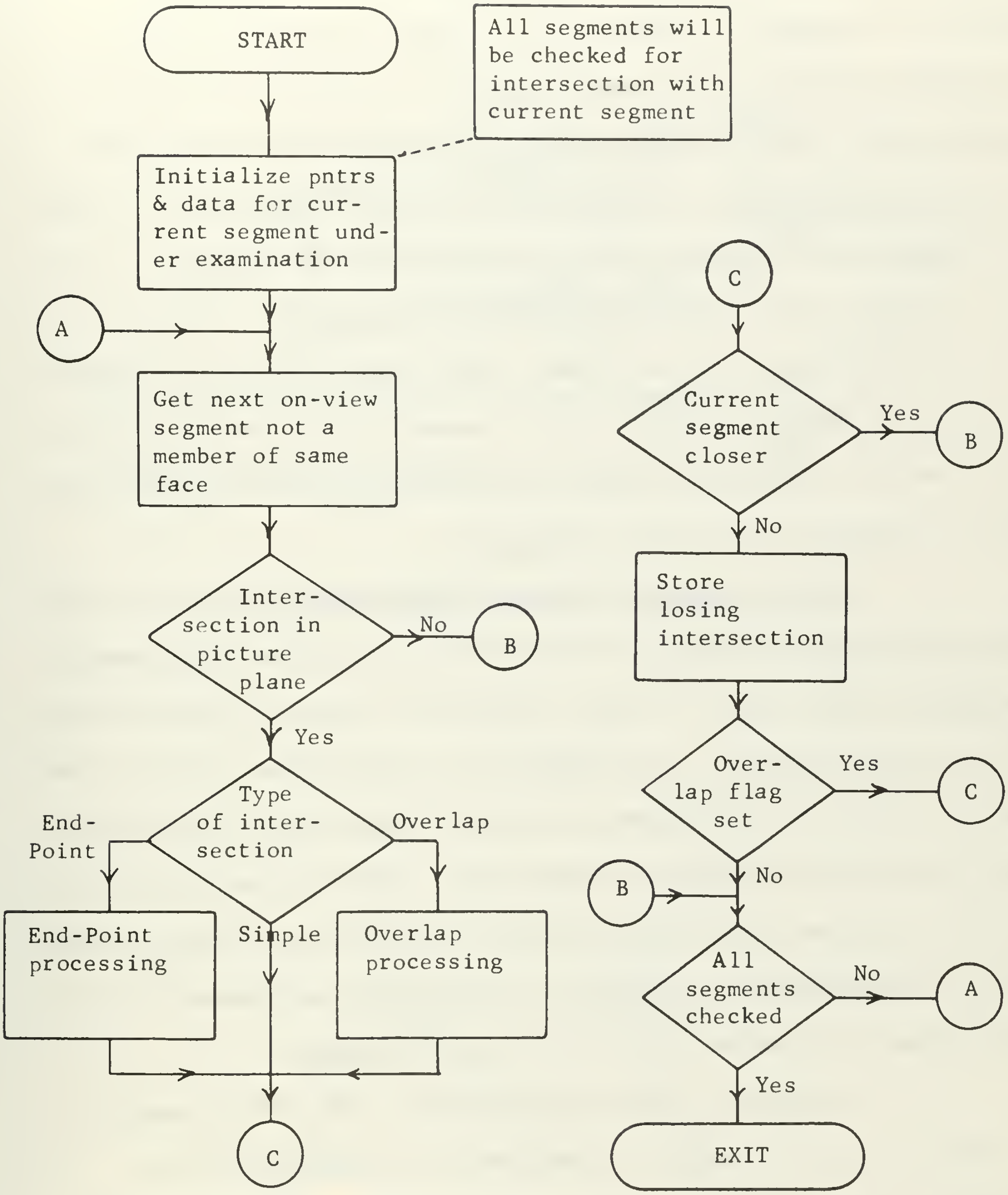
SUBROUTINE PLNLIN AND SUBROUTINE PIERC

the focal plane calculated in VMVMAT is used as one end point of the line. When the coordinates of the plane and the line have been determined, subroutine PIERC is called to determine the coordinates of the intersection between the plane and the line. These coordinates are used in the function DEPTH to determine the depth of the intersection. The depth of the node and the depth of the intersection are now compared and if the intersection is closer than the node, the face is added to LSTBFC. The number of such blocking faces is stored in the first integer of column 13 of ARRAY and the pointer to the starting entry for that node in LSTBFC is stored in the second integer of column 13.

b. Determination of Losing Intersections (LNTRSC)

During the examination of a segment, the initial vertex has its nature determined. Each time another segment crosses in front of the segment being examined (a losing intersection for the segment under examination), a change in the list of blocking faces occurs. The examination of a segment is done therefore by moving from the vertex with its nature known to each next intersection along the segment. Subroutine LNTRSC determines those intersections of direct concern (losing intersections) for the segment under examination (Flowchart 12). This subroutine uses the special test for implied vorticity previously discussed to determine whether or not an intersection has taken place.

Upon entering the subroutine, those quantities that will be needed but have not yet been determined for the current segment under examination are found. When all the characteristic quantities for the current segment have been determined, all segments (called test segments) are processed to determine if a losing intersection has occured with the current segment. Those segments which do not correspond to edges (i.e.


START
All segments will be checked for intersection with current segment
Initialize pntrs & data for current segment under examination
A
Get next on-view segment not a member of same face
Intersection in picture plane
No
B
Yes
Type of intersection
End-Point
Overlap
Simple
End-Point processing
Overlap processing
C
C
Current segment closer
Yes
B
No
Store losing intersection
Overlap flag set
Yes
C
B
No
All segments checked
No
A
Yes
EXIT

FLOWCHART 12

SUBROUTINE LNTRSC

lines lying on a surface) and those segments resulting from the intersection of two hidden surfaces are eliminated from consideration. However if a segment is hidden but is a member of an on view face (possible in the case of a concave dihedral), it must be checked for intersection since a change in the set of blocking faces occurs when such a line is crossed. In Figure 23, when the current segment is AB, test segments CD, EF, GH, IJ, KL, MN, OP, and QR cause changes in the set of on-view faces blocking AB when moving from A to B. Test segments CD, QR, and GH each add a face. Test segments EF and MN each add one face and subtract one face. Test segments IJ, KL, and OP each subtract one face. The set of faces is identified not only by number but also by identifying each face in the set.

Special attention is given to intersections at an end point (Figure 24) since the implied vorticity test as previously discussed is ambiguous in this region. If the intersection occurs at the end point of the current segment, it is treated as an intersection. If the intersection occurs at the end point of the test segment, the intersection point and the other points are "remembered". When a second intersection occurs at the same point, the other end of the new test segment is used with the remembered end-point value to construct a new segment which is then checked for intersection with the current segment.

In Figure 24, the current segment is AB. When segment CE is examined, an intersection is "remembered" for point E and the opposite end point C is remembered. When DE is examined, the intersection at E is detected. Since there is already an intersection at E, the opposite end point D and the stored end point C are tested as a new segment CD. In this case the constructed line results in no intersection.

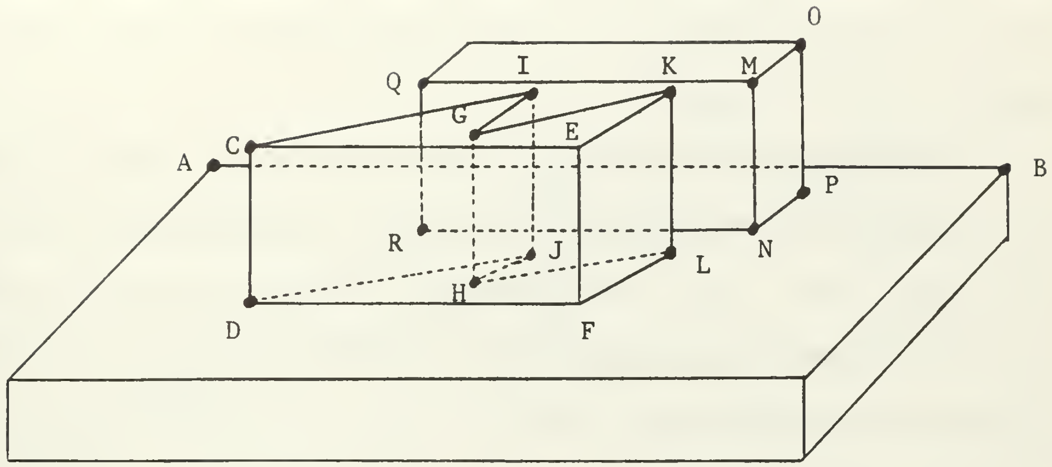


FIGURE 23

TEST SEGMENTS AFFECTING SET OF BLOCKING FACES

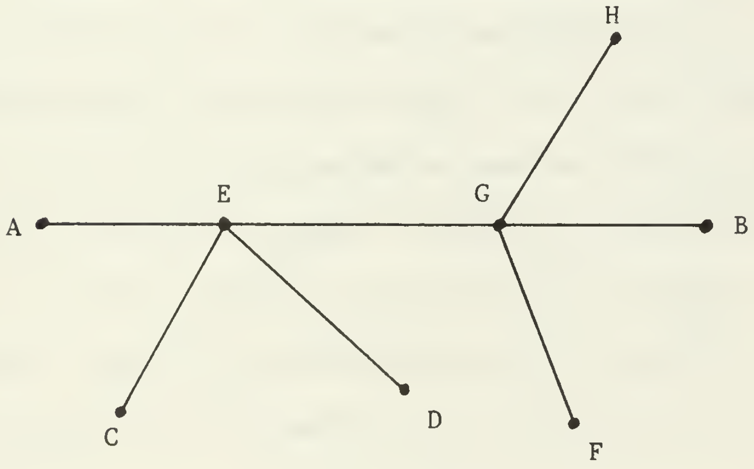


FIGURE 24

INTERSECTION AT AN END POINT OF A TEST SEGMENT

The same procedure is repeated for FG and GH, but in this case the constructed segment FH results in the detection of an intersection. Segment GH is then processed as in any other detected intersection. Segments CE, DE, and FG were treated as if no intersections had occurred (other than remembering the end points).

Before calculating the coordinates of the intersection of the two segments, a test is made to determine which segment lies in front of the other. Only the coordinates of those intersections where the current segment is losing need be calculated. Figures 25 and 26 show how this determination is made.

A sighting plane is constructed using the focal point and the two end points of the edge corresponding to the current segment. The observer "sights" along this plane when viewing the edge in space. The intersection (piercing point) of the sighting plane and the edge corresponding to the test segment is then determined through the special test for the intersection of a line and a plane. Another line is constructed in the sighting plane connecting the focal point and the piercing point. This line is then tested by the implied vorticity test for intersection with the edge corresponding to the current segment. If an intersection occurs, the current segment is winning because the test segment lies on the far side (from the observer's position) of the current segment. When no intersection occurs, the current segment is losing since the test segment lies between the observer and the current segment.

Figure 25 shows how the situation of one segment behind another can occur. Figure 26 is reduced to three segments and the focal point. AB is the edge corresponding to the current segment.

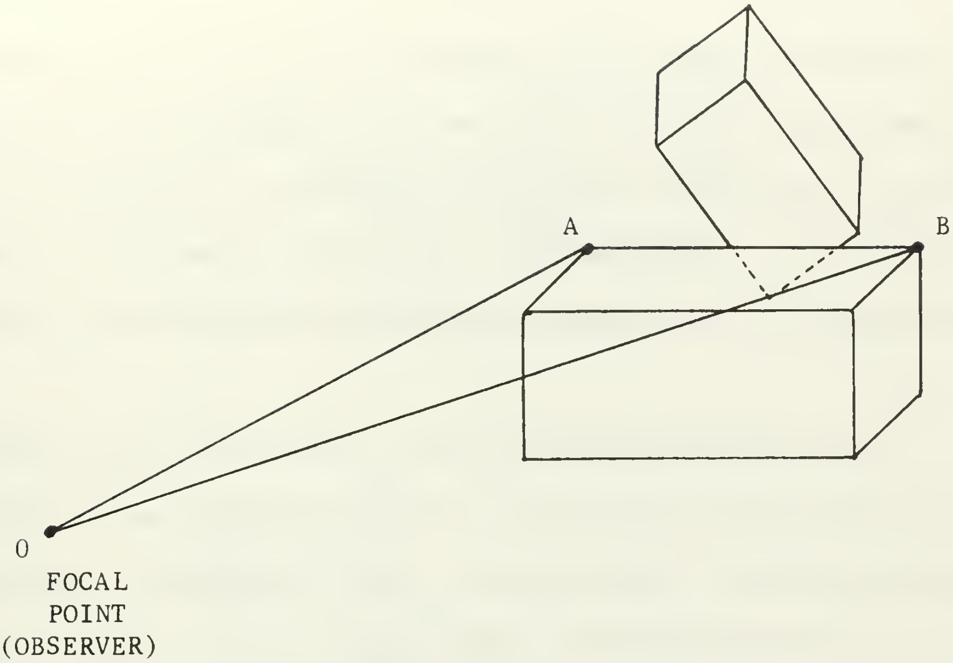


FIGURE 25

SITUATION REQUIRING DETERMINATION OF CLOSER LINE SEGMENT

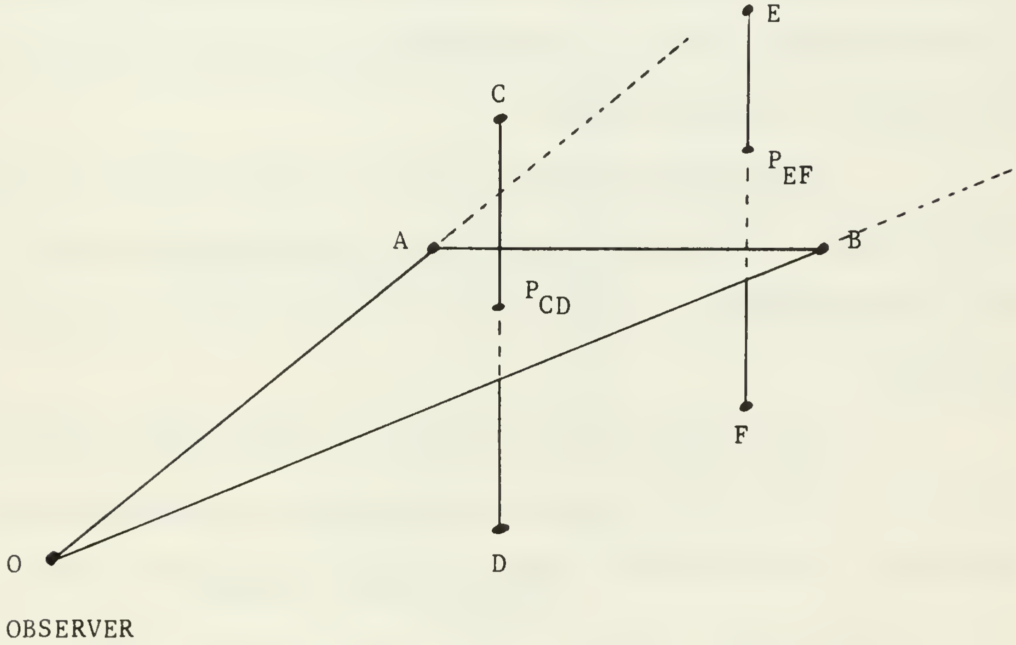


FIGURE 26

INTERSECTIONS IN THE SIGHTING PLANE

The sighting plane is determined in three dimensions by the focal point 0 where the observer is located and the end points of AB. The sighting plane is extended so that the intersection of EF and the plane can be seen. Exam segments CD and EF intersect the sighting plane at points P_{CD} and P_{EF}. Constructed line segment OP_{CD} does not intersect with the current segment AB, thus indicating that CD is in front of AB. However OP_{EF} does intersect with AB and this indicates that AB lies in front of EF.

The coordinates of the point of intersection can be converted to picture plane coordinates through the use of the homogeneous transformation matrix. The coordinates of the losing intersections are necessary to determine where the nature of the current segment changes. The point-slope form of the line equation in the two dimensions of the picture plane can also be used to determine where two segments meet at a common point. When using slopes the infinite slopes of vertical lines must be avoided. This is accomplished by assigning a large number as the slope of a vertical line.

Overlapping segments present a special case of multiple intersections. The various situations of overlapping segments were discussed previously in this chapter in the section on Implied Vorticity.

c. Determination of the Nature of a Near Point (TSTNNP)

On the rare occasion that a losing intersection occurs at one of the vertices of the current segment, it becomes necessary to be able to test a point a small distance away from the vertex to resolve the ambiguity shown in Figure 27.

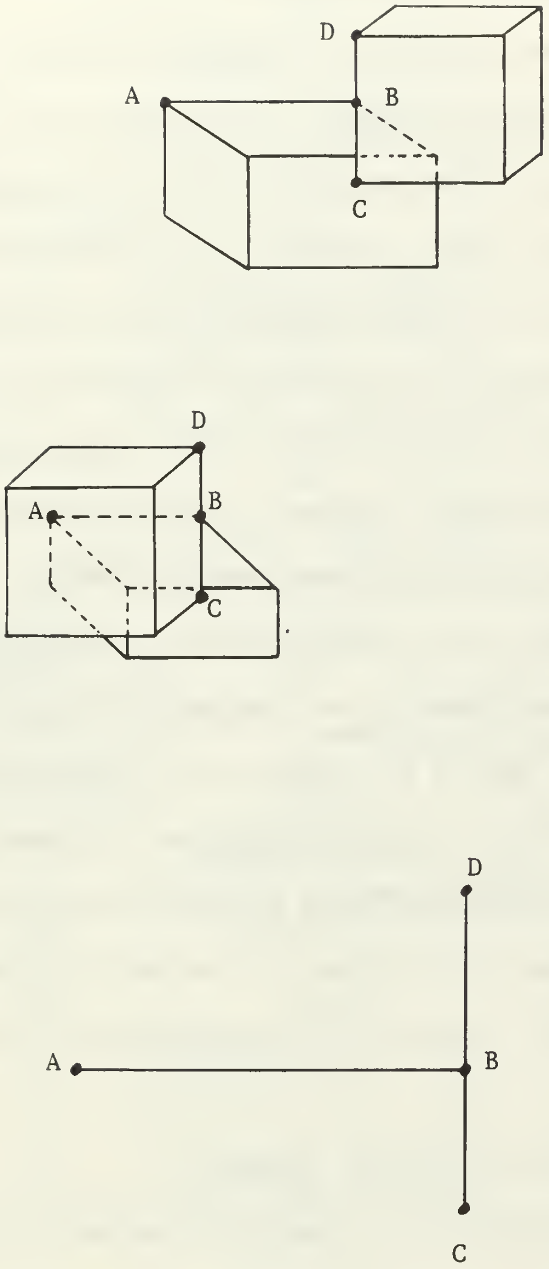


FIGURE 27

THE END POINT AMBIGUITY

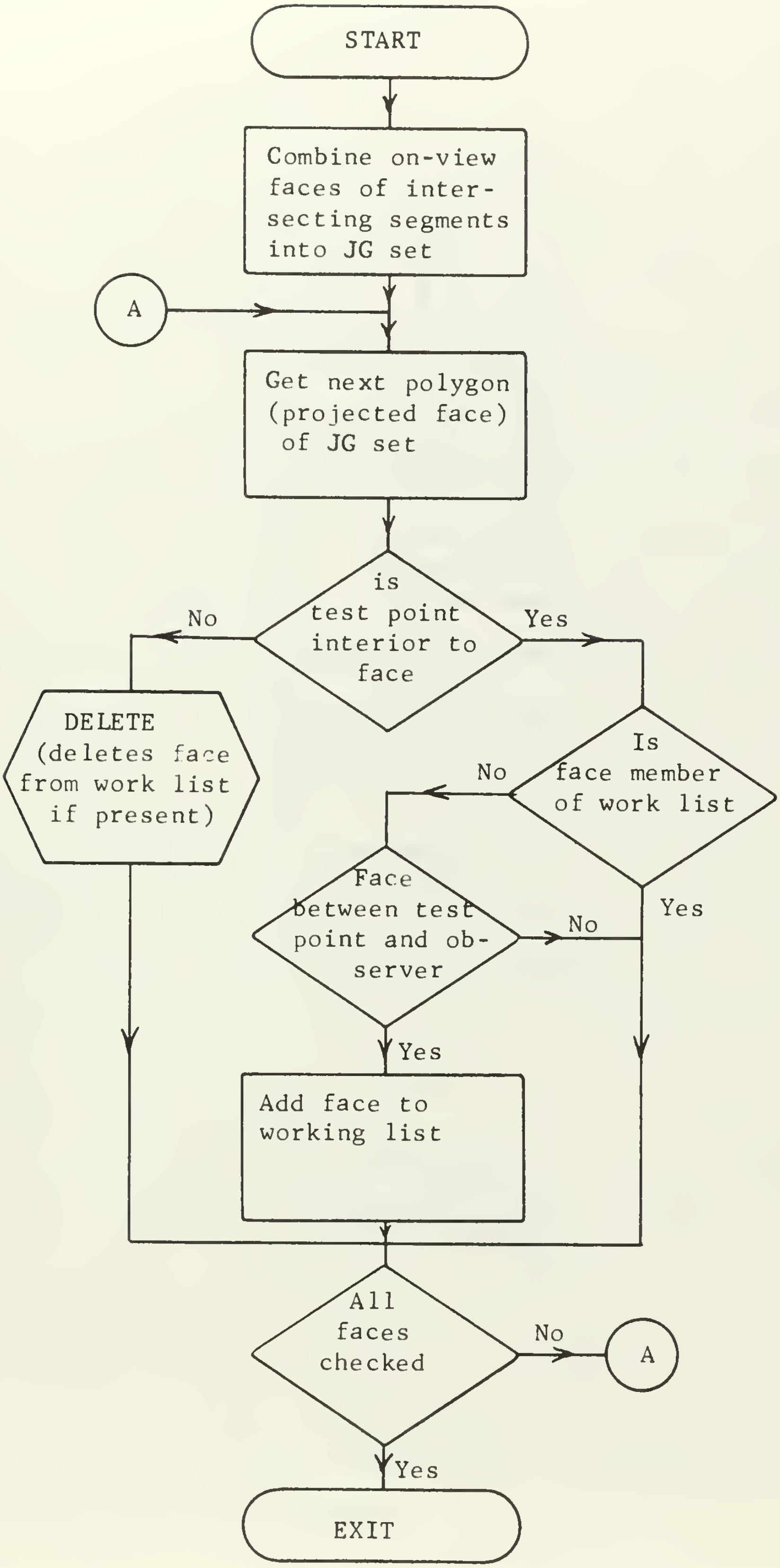
Here the current segment is line AB which intersects line CD at B. Assume B is losing with respect to line CD. The ambiguity here is whether line AB is going behind or emerging from the surface bounded by CD. TSTNNP resolves the ambiguity (Flowchart 13).

The subroutine (TSTNNP) is much the same as VRTNTR except that only a few faces need be checked. Since each segment has associated with it two faces, only a maximum of two faces need be checked for every intersection at the node. A set of faces belonging to all the intersections at the node is determined by combining the one or two faces for each intersecting segment (one face may be hidden). The nature of the specified point is then tested with regard to this set of faces in the same manner as VRTNTR tests all faces.

A working list of blocking faces is kept while examining a segment. Those faces found to be blocking the near point are compared to the working list of blocking faces. Faces which are a part of both lists remain. Faces in the working list not found while testing the near point are deleted and faces found in the test but not in the working list are tested to determine if the face lies between the test point and the observer. If the face blocks the point it is added to the working list.

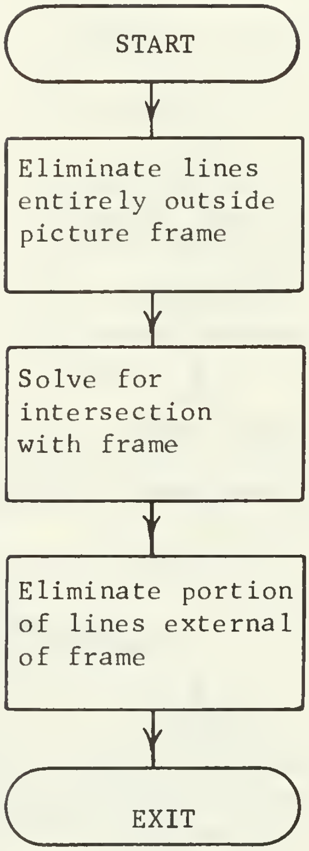
d. Scissoring the Picture (WINDOW)

As each end point is determined for the output line list, subroutine WINDOW (Flowchart 14) is called to "crop" the picture. Portions of the picture may extend beyond the normalized boundaries of the frame, and this cropping allows the portion of the picture within the frame to be drawn without distortion while eliminating that portion of the picture external to the picture frame.


START
Combine on-view faces of intersecting segments into JG set
A
Get next polygon (projected face) of JG set
is test point interior to face
No
Yes
DELETE (deletes face from work list if present)
Is face member of work list
No
Yes
Face between test point and observer
No
Yes
Add face to working list
All faces checked
No
A
Yes
EXIT

FLOWCHART 13

SUBROUTINE TSTNNP


START
Eliminate lines entirely outside picture frame
Solve for intersection with frame
Eliminate portion of lines external of frame
EXIT

FLOWCHART 14

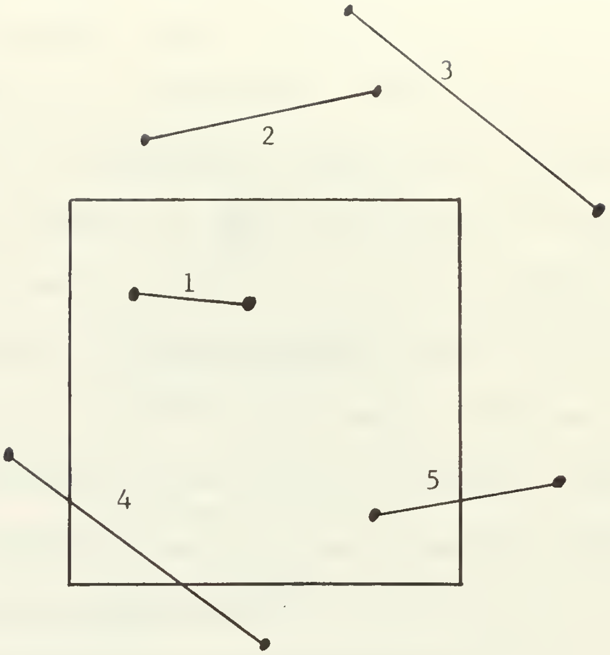
SUBROUTINE WINDOW

There are a number of cases possible for the position of any given line with respect to the picture frame. Figure 28 shows the possible cases with the line number corresponding to the case. WINDOW identifies which case exists and takes the appropriate action. Each line has a start and an end point which must be separately processed.

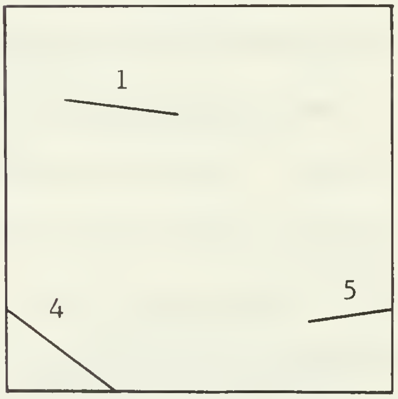
Line 1 requires no action. Both ends of the line are within the frame. Line 2 is the condition where the line lies entirely above (below, or to one side) of the picture frame. In this case the line is simply "erased". Line 3 also lies outside the picture frame but the detection of this condition is more difficult. Intersection with the vertical border is calculated and if the intersection exceeds the picture frame bounds, the intersection with the horizontal frame is calculated. If this also exceeds the picture frame bounds, the line is erased since no part of it can lie within the frame. Line 4 is treated as Line 3 except that in this case the intersection lies within one of the frame boundaries. The intersection of the line with the frame becomes the new end point of the line in the output line list. The procedure is repeated for the end point. Line 5 is similar to Line 4 except that one end point is already inside the picture.

When determining which lines lie entirely above, below, or to one side (case 2), both the start and end points are used. In case 3 the end point need not be checked since no intersection is possible, once it has been established that there is no intersection from the start point. In cases 1, 4, and 5 both start and end points are checked.

The result is that each line segment extending beyond the frame is cut off at the frame boundary.



BEFORE CROPPING



AFTER CROPPING

FIGURE 28

CROPPING THE PICTURE TO FIT THE FRAME

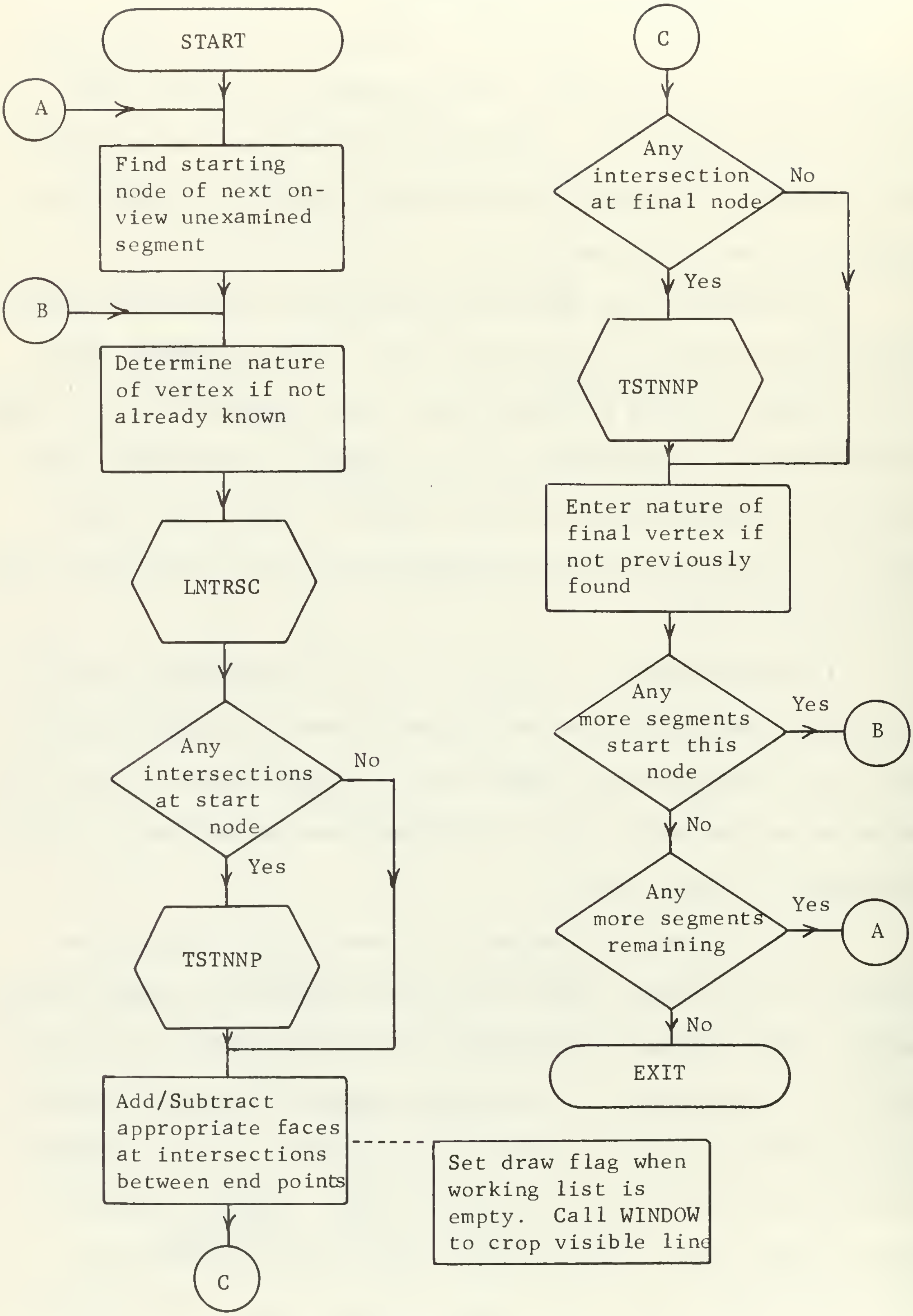
9. Examination of the Segments (CMPUT)

With all the preliminary steps completed, all that remains is to examine each on-view segment over its length to determine which portions of the segment are visible and therefore should be drawn and which portions of the segment are hidden and therefore not visible and should not be drawn (Flowchart 15).

To accomplish this examination methodically, the first on-view segment has the nature of the initial vertex tested by VRTNTR. Each on-view segment starting with this node is then examined in turn. Losing intersections are determined by a call to LNTRSC. If no losing intersections are detected and the initial vertex is visible, the entire segment is visible and the nature of the final vertex is the same as that of the initial vertex.

If intersections are detected, it must be determined if any of the intersections occurred at the initial vertex. If intersection did occur at the initial vertex, TSTNNP must be called to test a point near the vertex and the working list of blocking faces is modified to reflect the nature of the near point.

The draw flag is set whenever no faces block a segment from view and is reset whenever faces block the portion of the segment being examined. When setting the draw flag, the starting points for the output line are entered in the output list. When resetting the draw flag, the end points are entered in the output list after a call to WINDOW to crop the line to fit into the picture frame. (However, when the test of a point near the initial vertex causes the draw flag to be reset, the initial point of the line is deleted since no line exists.)


START
A
Find starting node of next on-view unexamined segment
B
Determine nature of vertex if not already known
LNTRSC
Any intersections at start node
No
Yes
TSTNNP
Add/Subtract appropriate faces at intersections between end points
Set draw flag when working list is empty. Call WINDOW to crop visible line
C
C
Any intersection at final node
No
Yes
TSTNNP
Enter nature of final vertex if not previously found
Any more segments start this node
Yes
B
No
Any more segments remaining
Yes
A
No
EXIT

FLOWCHART 15

SUBROUTINE CMPUT

The segment under examination is now traversed to the point of the first intersection. If no more intersections occur, the segment is drawn or not drawn according to the working nature. The number of intersections at the final vertex is detected and at any time that the number of intersections remaining equals the number of intersections at the final vertex, there are no more intervening intersections and the final vertex can be tested by TSTNNP for a change in its nature.

The intervening intersections are used to change the nature along the segment by using the one or two on-view faces associated with each intersecting test segment to modify the working list of blocking faces. If the list contains a face that is one of the two defining faces of a segment, that face is deleted from the list. If the list does not contain the face, the face is added to the list. (See LNTRSC for an example.)

If more than one intersection occurs at the same point, this fact must be detected. Three cases are possible when more than one intersection occurs. Figure 29 shows the three cases. The intersections occur on the picture plane and in each example the intersection is losing for the isolated segment.

The intersection at P_1 occurs because two other segments also intersect at P_1. All three of the edges are at different distances from the observer. These intersections are processed as two different intersections that just happen to both occur at P_1.

The multiple intersections at P_2 and P_3 occur because the point of intersection is a corner, itself the intersection of three edges. Segment CD and one of the segments of the corner intersection overlap between the intersections. Both faces defining the edge corresponding

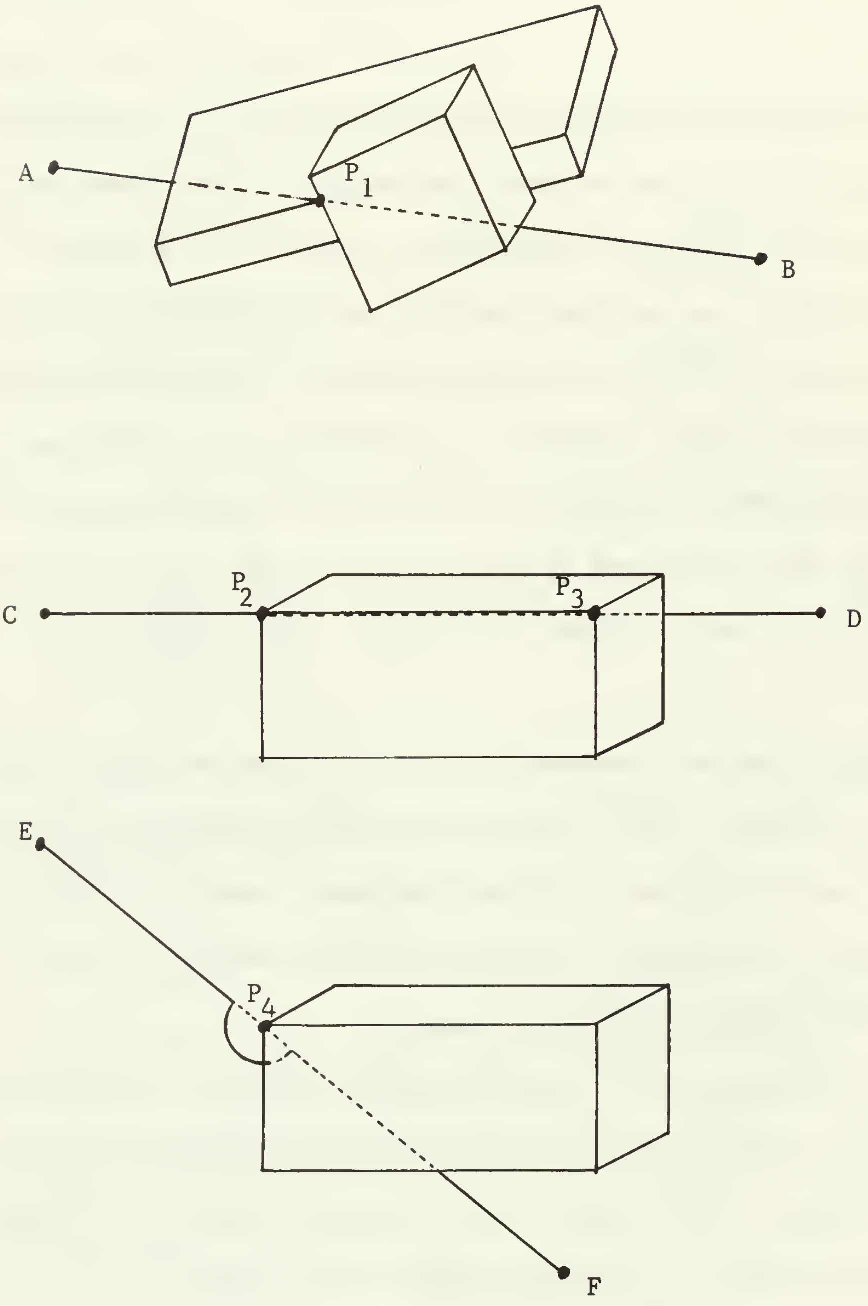


FIGURE 29

SPECIAL CASES OF MULTIPLE INTERSECTIONS

to the overlapping segment are added to the working set of blocking faces when arriving at P_2 from point C. Both faces are subtracted when leaving P_3 and proceeding towards D.

The multiple intersections at P_4 occur because the point of intersection is again a corner. This case differs from the previous example in that segment EF does not overlap any of the other segments. To determine which face or faces should be added, a semi-circle is constructed around the intersection (either direction is equally satisfactory) and intersections with the semi-circle are tested as single intersections. The semi-circle is in effect constructed through the use of the implied vorticity test and restricting the intersections to one side only.

When all intersections before the final vertex have been processed, only the final vertex remains. If no intersections occur at the final vertex, the examination of the segment is complete. If intersections occur at the final vertex, a point near the vertex has a nature already known. TSTNNP is then used to find the nature at the vertex. The nature of the final vertex is entered in column 13 of ARRAY if not already known.

The next segment starting with the same node is then processed or if no other segments start with this node, the next on-view segment in the list is processed until all segments have been examined.

10. Preparation for a New View (ERASE)

Before processing a new view, column 13 is reset to zero for both integer values. This is necessary because the contents of column 13 indicate whether or not the nature of the associated node is known. The negative sign in the second integer of column 9 which indicates that the segment has been examined is removed. If ALTER changed the data structure, ERASE calls subroutine LINES to restructure the original line list.

E. SUMMARY

The hidden-line removal algorithm presented here is flexible and general so that it is easily tailored to specific applications. The algorithm could be easily adapted for on-line structure modification or it could be adapted for rapid calculation for display of a restricted set of views. A single object with no concave dihedrals that is always in front of the observer can be displayed after processing for hidden surfaces (HDNSRF) by connecting the end points of the visible lines and cropping those lines.

The entire algorithm has been committed to a FORTRAN IV program. The algorithm holds whether it is committed to software or hardware or to a combination of the two. If assembly language were used, the greater flexibility would allow flags to be used more effectively and to a greater extent. However the ideas advanced by the algorithm hold regardless of the method of implementation.

VII. SUMMARY

An introduction into the basic concepts and requirements encountered in the field of three-dimensional computer graphics has been presented. Techniques were developed for presenting the display in two dimensions of real-space three-dimensional objects. Transforming the coordinates of real space into two-dimensional perspective space was the subject of chapters II and III. Chapter IV introduced the requirements for structuring data to fit the application at hand, beginning with the simple case of two-dimensional coordinate storage and advancing to the three-dimensional case. Since contour lines are relative to the field of computer graphics in three dimensions, chapter V introduced basic concepts that may be the basis for further study and development. A study in three-dimensional computer graphics is not complete without an investigation into the removal of hidden lines. Chapter VI presents an algorithm for the elimination of all lines (or those portions of a line) not visible to the observer.

A. THE TRANSFORMATION

The computer can be utilized to obtain perspective views of three-dimensional objects from any specified angle and position. The three dimensions of the real world are transformed into the two dimensions required for a drawing on a plane surface. To minimize the number of points that must be transformed, it is convenient to approximate an object under view by line segments corresponding to the outline contours of the object. If the object is constrained to be approximated by intersecting plane surfaces, the line segments will all be straight lines.

A line is then transformed by transforming the end points and connecting the transformed points by a straight line.

The transformation of a point is accomplished by post-multiplying the coordinate vector of a point by a 4 x 4 matrix. A fourth homogeneous coordinate (equal to 1.0) is added to the coordinate vector to make the multiplication compatible. The time required to transform a picture is thus dependent only on the number of defining points in the picture. The conversion time was approximately 600 microseconds per point after the matrix had been determined. (All times are approximate and are in reference to the IBM 360-67 computer located at the Computer Facility, Naval Postgraduate School.) The calculation of the transformation matrix is independent of the picture complexity and is obtained as the product of two rotation matrices. Direction cosines are used to determine the elements of each of two rotation matrices. The time required to determine the entire transformation matrix was 20 milliseconds. Associated with the computation of the matrix is the determination of the location of the focal point and three points in the focal plane in terms of the coordinate system used to describe the object under view. The time required for this computation was approximately 16.5 milliseconds. These positions are needed in the processing for the removal of hidden lines.

B. DATA STRUCTURE

In the field of three-dimensional computer graphics, data can be stored in many different ways. There is no one best way. For a specific application one method may be better than another but many of the same elements exist in all methods. Chapter IV discussed requirements in storing data and some methods to meet the requirements. Two requirements which are generally conflicting are that the data be readily available and that the data be efficiently stored.

In storing the data for the developed program three factors were considered. Entering data must be a simple process and must not be confusing to the user. The data must be available quickly and the data must be compactly stored. Compromise on all three requirements had to be made. Accessability of the data was compromised the most to keep from imposing even greater dimensioning requirements on the user. The subroutine FACES causes a search to be made of existing points to determine if that point is already entered . Thus as the number of points already entered increases, the time to enter a new point also increases. For the 12 point L-shaped block example in the computer output section (Figures 32, 33, and 34) the time for entering the points was approximately 0.158 seconds. For the 62 point aircraft carrier (Figures 35-39) the time for entering points was 1.07 seconds. The local computer system does not read directly from the card reader but instead loads the output of the card reader on discs for processing. The above times therefore do not reflect access time to the card reader.

When the points and other necessary data have been entered for all the faces, LINES is called to construct the line structure in storage. Each new line that is to be entered is first checked with all lines already entered so that the same line will not be entered twice. As the list of lines grows, so does the search time.

When all lines have been entered, each line is checked to determine if it is a concave dihedral. CCVCVX does this to complete the "permanent" structure of the object in the data array. CCVCVX calculates normals for each surface and therefore is in part dependent on the number of faces.

Each line is then checked so that the time for the second part of this subroutine is dependent on the number of lines. The time to complete the permanent structure through the combination of LINES and CCVCVX is 39 milliseconds for the L-shaped block and 0.627 seconds for the aircraft carrier. The L-shaped block is composed of 32 surfaces and 93 lines.

The object is now fully described in the data array and is completely independent of any view specified (with the exception of having a portion of the object behind the viewer).

C. HIDDEN-LINE ELIMINATION

The hidden lines are removed in two steps. In the first step, HDNSRF detects those surfaces which face away from the observer. Lines on these surfaces which cannot be seen by the observer are eliminated. All lines thus hidden by the object's own volume are eliminated. Each projected face must be checked, so the process is dependent in part on the number of faces and in part on the number of lines that must be checked. The L-shaped block is processed in 5.5 milliseconds and the aircraft carrier is processed in 24 milliseconds for this portion of the hidden-line removal.

Alteration of the structure to terminate lines in the focal plane takes 2 milliseconds for the L-shaped block and 8 milliseconds for the aircraft carrier when no change is made. Each visible line must be checked to determine whether or not it should be terminated. In the case of the fifth view of the carrier (Figure 39) where the aircraft has landed on the deck, eight lines were terminated with new points lying in the focal plane. The total time for ALTER in this case was 18 milliseconds.

The second step in removing hidden lines is done by examining each segment not yet eliminated over its length and drawing that portion which remains in view. During the process of examining a segment, the intersection with every other segment must be determined. Only those intersections which affect visibility of the segment under examination are noted. This results in the test being done nl (number of lines) times for each segment. The test must be made for all segments not yet eliminated. Additionally the nature of the starting vertex must be determined as well as the nature of any vertex not found during segment examination. The time necessary to determine the nature of a vertex is dependent on the number of lines also. The time required for the subroutine CMPUT is therefore related to the square of the number of lines in the structure. This is only an approximation since the path taken is view-dependent. The L-shaped block is processed in an average of 77 milliseconds and the aircraft carrier processing time varies between 1.5 seconds and 2 seconds, depending on the view. The program can be modified at the expense of storage to allow intersections of the intersecting segments to be saved and thus testing half as many lines. By allowing reversed examination of a line, some calls to determine the vertex nature could be eliminated.

D. THE COMPUTER OUTPUT

Three views of an L-shaped block are given in Figures 32-34 in the computer output section. Five views of an aircraft carrier are given in Figures 35-39. The pictures were drawn by the CALCOMP Plotter at the Computer Facility, Naval Postgraduate School. Figures 36-39 form a sequence as if a high speed aircraft were approaching for a landing, and in the final view (Figure 39) the aircraft is on the deck.

Associated with each Figure is a set of data making up a table. Each table has five parts. Part (a) is a listing of the parameters for the view following. Figure 30 is a two-dimensional plot of position for the landing approach shown in the last four views of the carrier. Part (b) of each table is the listing of the nodes and associated information of the data array. Part (c) is a list of the lines and part (d) is a pair of lists, LSTFAC and LSTBFC. LSTFAC lists the points in the order entered (reading from left to right). LSTBFC lists the faces blocking various nodes. Pointers in the node data point to LSTBFC. Part (e) of each table is a listing of the output line list available for use by any output device.

E. CONTOUR LINES

The treatment of contour lines presented in chapter V was included because contour lines are very much a part of three-dimensional computer graphics. Contour lines have not received a great amount of attention because it has been easier to avoid the problem than to develop techniques for working with such lines. Display of contour lines has also been a problem. With the ability to obtain perspective transformations the display problem becomes one of finding a convenient way of handling the lines. When techniques are developed, the contour lines can be displayed in perspective with hidden lines removed as with the plane-surfaced objects used here.

F. APPLICATIONS

The potential applications of the perspective transformation and the program for hidden-line removal are many. Two examples are given in the computer output section. In one case just a simple display of an object

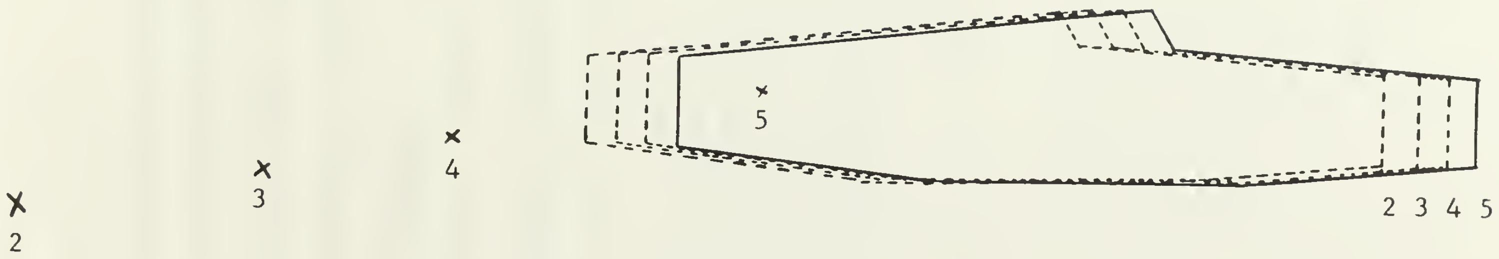


FIGURE 30

POSITIONS FOR CARRIER VIEWS (2-5)

from various aspects was shown. The second case allows movement of both the object under view and the observer.

The program as developed may be used to display any three-dimensional object or structure that can be described by plane surfaces. The crystal lattice structure of a semiconductor could be shown as easily as the view of the aircraft carrier. The focal length would have to be changed to correspond to the size of the picture being taken. The print of the picture being taken will always be the reciprocal of the scale factor times 1/2 in the units used to measure the focal length. Therefore if it is desired to view data measured in microns, the focal length should be in microns.

The computer not only acts as a camera, but acts as a variable size camera where both the lens and area of the focal plane can be varied. The computerized camera is completely mobile and will present a picture from the specified orientation according to the data that was entered into and resides in the data array.

G. DEVELOPMENT

The elements for perspective display have been presented. Additional work may extend the developed programs in a number of ways.

1. Simulators Using Displays

For specific applications, the program as developed can be tailored to operate at faster speeds. This is especially true if the program or a part of it is committed to assembly language. Additional flags can identify certain lines that need not be checked. Additional lists in memory can speed the search for points or lines. By bit manipulation, the list of blocking faces can be disposed of and the information can be stored in just a few computer words where each bit identifies one face. Special-purpose computers can speed such a system where logic choices determine paths and where logic choices are based on comparisons. If one section of a computer can handle the refresh requirements of the display system and another section can update in fractions of a second, an apparent real-time system could be developed. Small angular changes in pitch, roll, and yaw can be handled by shifting or rotating the picture already formed. For small angles the results are the same as recalculating the entire picture. These quickened responses could be superimposed on the calculated picture between computations of new views.

2. On-Line Computer Graphics

The elements contained in this study can be applied to modification on-line of the item under view. The feature of adding surfaces already exists, though a modification would be desirable to keep LINES from redrawing already established lines. A simple flag should remedy that situation. Deleting surfaces would take additional work in identifying the surface and lines to be removed and in closing up any holes left.

3. On-Line Authorship

Just as text material can now be written on line and formatted by the computer, the insertion of pictures with the text material would be a tremendous aid when developing new material. The writer could have his drawings done while the ideas are fresh in mind and formatted to proper size by the computer.

4. Extensions to Curved Surfaces

The next class of objects in the hierarchy of complexity are those defined by curved surfaces. An extension to include the curved surface is a natural step although a formidable one.

5. Tactical Displays

Three-dimensional displays may have significance in displaying tactical situations. In this case, some study must be extended towards the determination of the best method of representing the information without confusion. The ability to move around in space allows a study of the tactical situation from different aspects.

6. Display of Surfaces as a Function of Two Variables

If the problem of handling contour lines can be resolved for large numbers of points, surfaces which are the function of two variables could be studied with greater effect than now possible. This is an area where much work still needs to be done.

H. FINAL

Throughout each application the basic transformations have been on points which in turn determine the lines of the outlines. Transformation of points is simple and fast. However the connection of the points is less simple and becomes time consuming as picture complexity increases. Hardware and assembly language can reduce the time required. The complexity is overcome by following the algorithm presented although processing time will increase approximately in proportion to the square of the number of lines defining the object.

APPENDIX A

GLOSSARY

ALPHA	The angle of rotation, u into v about the w axis of the object. This corresponds to the object's yaw.
BETA	The angle of rotation of w into u about the v axis of the object, corresponding to the object's pitch.
COMPUTED NODE	All visible segments from this node have been examined.
DIRECTION COSINES	Cosines of the angles that the new axes (after rotation) make with the old axes (before rotation). These are different for the two cases of either Euler Angles or Fixed Angles.
DISPLAY COORDINATES	Division of the second and third perspective homogeneous coordinates by the fourth coordinate results in the horizontal and vertical display coordinates respectively.
EDGE	The intersection of two faces in three dimensions. The two-dimensional equivalent is the segment.
EULER ANGLES	The axes of an object turn with the object and all rotations are then about the new orientation of the axes. The specified order of rotation for this study is Alpha, Beta, and then Gamma. This order agrees with the use of Euler angles in aircraft coordinate systems.
EXAMINED SEGMENT	The segment has been examined for all intersections and drawing information for that segment has been computed.
FACE	The portion of a plane bounded by the edges and used to construct some object in three dimensions. The two-dimensional equivalent is the polygon.
FIXED AXES ROTATION	The axes of rotation do not turn. All rotations are about fixed axes which will not move.
FIXED INERTIAL REFERENCE	With no movement or rotation of either the viewing plane axes or the object axes, the object axes and the viewing plane axes correspond and define the fixed inertial reference.
GAMMA	The angle of rotation of v into w about the u axis of the object, corresponding to roll of the object.

HOMOGENEOUS COORDINATES	Real-space coordinates are multiplied by some factor which then becomes the fourth homogeneous coordinate. In this floating-point implementation the coordinate is set to 1. For fixed-point implementation this factor can be a normalizing factor for each point.
HOMOGENEOUS TRANSFORMATION MATRIX	The 4 x 4 matrix which can be used to transform any four-dimensional homogeneous coordinates to perspective homogeneous coordinates. This matrix combines rotation, movement, perspective, scaling, and offset transformations.
NATURE	The set of faces that obstruct a node from sight.
NODE	A three-dimensional point, normally the intersection of two or more edges. The two-dimensional equivalent is the vertex.
OBJECT AXES	The coordinate system in which the object is referenced. The object is that which is to be displayed. The axes are labeled u, v, and w.
OBJECT AXES MOVEMENT	The movement of the object axes in respect to a fixed inertial position.
OFFSET	The distance from the center of the whole picture to the center of the enlarged picture.
ORIENTATION OF THE VIEWING PLANE	The angular rotation from a fixed inertial reference.
PHI	Angular orientation of the viewing plane about the x axis.
PSI	Angular orientation of the viewing plane about the z axis.
SCALE	The distance from the center of the picture to one edge. A normal picture (no magnification) has a scale corresponding to 1/2. A scale of 1/4 would be a magnification of 2.
THETA	Angular orientation of the viewing plane about the y axis.
VIEWING PLANE AXES	The coordinate system based on the viewing plane, such that the picture is contained in the yz plane and the focus is on the positive x axis so that the viewer looks into the negative-x half space. The observer's position will always be the focal point.
VIEWING PLANE MOVEMENT	The movement of the viewing plane from a fixed inertial position.

WIRE FRAME	A drawing showing all the lines whether or not those lines are hidden. The drawing appears as though the object is made of wires.

APPENDIX B

FORTRAN IV PROGRAMMING DIFFERENCES

Although the attempt was made to program the IBM 360-67 and the SDS-9300 as nearly alike as possible, some differences were necessary due to the individual characteristics of the two computers. The properties discussed here are those which caused differences in programming the two computers.

A. INTERNAL SUBROUTINES VERSUS ENTRY POINTS

On the IBM 360-67, entry into a subroutine is allowed through the use of an entry statement. The statement takes the same form as a subroutine statement except that ENTRY is used in place of SUBROUTINE. Arguments are allowed if desired. However, no subroutine may be called by itself through any of its entry points (not reenterant). ENTRY statements encountered during the execution of a program are treated as CONTINUE statements.

On the SDS-9300, no ENTRY statements are available but internal subroutines are allowed. Internal subroutines are similar to the usual subroutine except that no END statement separates those subroutines internal to a larger subroutine. During execution, the program is not allowed to run through any SUBROUTINE statements.

The use of the entry points and internal subroutines allows the named variables to be available anywhere in the group of internal subroutines. This allows entry into the various routines without the need for a long string of calling arguments. A blank or named common would have to contain many variables to be used for this same purpose.

In both computers the main subroutine must be called before calling an entry point or an internal subroutine. The entry points on the IBM computer were treated as internal subroutines in this application.

Since the IBM-computer program cannot enter any of its own entry points, those subroutines which are called while executing an internal subroutine must be placed in a second subroutine with internal subroutines (entered by ENTRY statements). Thus the subroutines in QSCRB1 are called by the internal subroutines of DESCRB in the IBM-computer program. This was not necessary for the SDS computer program.

D. DO LOOPS

DO loops on the SDS-9300 computer are tested before entering and if the index value is greater than the terminal value (positive incrementing) the loop is not executed at all. In the case of the IBM computer DO loops are always executed at least once since the index is not checked until the end of the loop. This made it necessary to insert protection statements wherever the possibility existed that the terminal value might be less than the index value.

C. WORD LENGTH

Integers on the SDS computer are each one computer word (24 bits) long. Real numbers use two computer words and double precision uses three computer words. The IBM machine uses four bytes for both real numbers and integers unless otherwise specified. Specifying INTEGER*2 will cause the specified quantities to be represented as two-byte integers. However no library functions can be used with two-byte integers. The variables in two bytes must first be converted to four-byte integers before even the absolute value can be found.

D. MULTIPLE ASSIGN STATEMENTS

The SDS computer allows the use of multiple assignments such as

$$A = B = C = 0.0.$$

When convenient, these multiple-assign statements were used in programming the SDS-9300 computer.

APPENDIX C

USING THE PROGRAM

Although there are differences in coding between the computer programs due to the differences discussed in Appendix B, the use of the programs is quite similar. Since only minor differences must be taken into account, this discussion will apply to both computers. The exceptions will be noted.

The computer programs for both computers may be found following the Appendixes and preceeding the Bibliography. These may be referred to for illustrative purposes.

A. DIMENSIONING REQUIREMENTS

The user must dimension the following arrays with dimensioning statements similar to those listed.

```
DIMENSION ARRAY(N,13), IARRAY(2,N,13),FNORML(N,3)
DIMENSION LSTFAC(NT), LSTBFC(1)
DIMENSION XSTART(NL), YSTRAT(NL), XEND(NL), YEND(NL)
```

All of the integer variables must of course be replaced by the appropriate integer. The integers are defined as minima, and may be larger without any degradation in performance.

N is equal to or greater than the number of different points used to describe the object.

NT is equal to or greater than the total number of all points entered. Every point is counted in this case each time it is entered. For example, if the object were a cube bounded by six faces with each face determined by four points, the total number of points entered would be 4 x 6 = 24, although N, the number of different points would be 8.

If however, part of the viewed object will be behind the observer, N must be increased by one for each line that passes through the observer's plane since additional points are computed to determine where each line should terminate.

NL is equal to or greater than the number of lines that will be used to draw the finished picture.

For the IBM 360-67, IARRAY must be specified to be a two-byte integer array (i.e., INTEGER*2 IARRAY). IARRAY must overlay ARRAY such that two words in IARRAY occupy the same length as one word in ARRAY.

Two EQUIVALENCE statements are required. The statement

```
EQUIVALENCE (ARRAY(1,1),IARRAY(1,1,1))
```

will properly align the real array and the integer array. This statement is mandatory in this computer program.

The second EQUIVALENCE statement

```
EQUIVALENCE (FNORML(1,3),LSTBFC(1))
```

is used to conserve space by using a portion of the FNORML array for different purposes in different sections of the computer program. If this EQUIVALENCE statement is not used, LSTBFC must be dimensioned large enough to hold all of the blocking faces (a number which varies with the object and the view). The value of NF should generally be large enough for the array LSTBFC.

B. NAMED COMMON

The program requires the use of a named common which holds the angles and positions of the object and of the observer, as well as certain other information.

The COMMON statement is

```
     COMMON/SDFB/ALPHA,BETA,GAMMA,PSI,THETA,PHI,OBJMV(3),
    1VIEWMV(3), FOCUS,YOFF,ZOFF,SCLE,HMATRX(4,4),OBSPOS(12)
```

where

ALPHA is object rotation about a vertical axis.

BETA is object pitch.

GAMMA is object roll.

PSI is observer rotation about a vertical axis.

THETA is observer pitch.

PHI is observer roll.

OBJMV is an array containing the three inertial coordinates of the origin of the object axes in an inertial frame.

VIEWMV is an array containing the three inertial coordinates of the origin of the observer's axes in an inertial frame.

FOCUS is the focal length used to 'take' the picture.

YOFF is the horizontal offset from the normal center of the picture.

ZOFF is the vertical offset from the normal center of the picture.

SCLE is the normalized distance from the center to the edge of the picture. With no offset this distance is 1/2.

HMATRX is a 4 x 4 matrix which is used to transform each point to its value in the picture plane. This matrix is calculated once for each view.

OBSPOS is an array which contains the locations of three points in the observer's focal plane expressed in the object coordinate system. These positions are calculated once for each view.

C. EXTERNAL STATEMENT

The name of the subroutine which will be used to enter the object description will be passed as an argument in a call to subroutine DESCRB. The name therefore must be declared EXTERNAL. If the provided subroutine is to be used, the statement is

```
EXTERNAL READCR
```

and this statement should appear before any executable statements.

D. INITIALIZATION

In the case of the SDS-9300, internal subroutines are used to take advantage of access to variables in a number of the internal subroutines. The IBM 360-67 program utilizes entry points to accomplish a similar characteristic. Both computers require an initial call to identify the arguments. This call need only be made once while using the program. Since the initialization procedure resets many flags and pointers, the call to DESCRB should only be made when a new object is to be described. Previous descriptions are lost.

```
 CALL DESCRB(ARRAY,IARRAY,FNORML,LSTFAC,N,READCR,
1XSTART, YSTART,XEND,YEND,NLINES,LSTBFC,NUSED)
```

All of the arguments have been previously introduced except NLINES and NUSED. NLINES is the number of lines drawn for the completed view and is less than or equal to NL. NUSED is the number of different descriptive points identified by the program and will be less than or equal to N. Both NLINES and NUSED are calculated within the program.

Only that portion of ARRAY that needs to be cleared will be set to zero. ARRAY may be completely cleared before starting if desired, but it is not required. DESCRB will initialize pointers and set the angles defined above to zero. Positions of the object axes and the observer's axes will also be set to zero with one exception. The object will be moved 40 units away from the observer plane (focal plane). The focal length will be set at 1.5 units and the scale will be set to 1/2. Offsets will be set to zero. Any desired changes in these values should be made after the initial call to DESCRB.

The observer looks into the negative half-space. Thus the object is initially placed at (-40.0, 0.0, 0.0). The remaining positive directions are to the observer's right and up respectively.

E. DESCRIBING THE OBJECT

The object must be described by bounded plane surfaces called faces. Each face is described by listing the points which determine the bounds of the surface. The points must be listed so that the surface is always to the left when traveling from point to point. The first point for each face must be a corner point since the last point, the first point, and the second point are used to calculate an inward pointing normal and therefore cannot be allowed to be collinear. In each face each point is connected to the preceeding point and the final point is connected back to the first point.

To enter the points of a face, the calling statement is

```
CALL FACES
```

and this statement must be used for each face. It is convenient to use a DO loop to enter the faces.

```
   DO 1 J=1,NFACES
   CALL FACES
 1 CONTINUE.
```

Of course NFACES must be an integer number or must be defined before invoking the DO loop. Each face is then entered in turn using the subroutine specified in the initializing call to DESCRB. If a subroutine other than READCR is used, the subroutine must have dummy arguments that correspond to the arguments used in READCR to be compatible with the program. Two arguments are necessary. The first is an integer variable and the second is a single-subscript array of real numbers dimensioned for three coordinates. (DIMENSION TEMPIN(3) is an example.) On the first call to the subroutine, the number of points contained in the face is returned in the first argument. The succeeding calls return the three-dimensional coordinates of one point for each call where

```
TEMPIN(1) = u
TEMPIN(2) = v
TEMPIN(3) = w
```

in the object coordinate system. The subroutine will be called to read the point coordinates until the specified number of points have been read.

If the subroutine READCR is used, the data will be read by the card reader. The first data card for each face contains the number of points in that face to follow. This integer must be right justified to column 10. Each succeeding card contains the three coordinates of one point. The FORMAT expected for the coordinate points is 3F20.6.

F. COMPLETING THE DESCRIPTION

When all the faces have been entered, the points within each face are connected by

```
CALL LINES
```

followed by a subroutine to identify the concave and convex dihedrals.

```
CALL CCVCVX
```

This completes the structure within the computer. These subroutines are not called again until the object to be displayed is changed. As many views of the object as desired may now be developed by calling the succeeding subroutines.

G. DEVELOPING THE PICTURE WITHIN THE COMPUTER

Before continuing with the computation of the picture from the stored structure, any desired changes in values of the angles or positions of the object and the observer must be entered. Changes in focal point, offsets, and scale should also be made at this point. When these variables are set to the desired values (if not set, these variables remain set to the last value used, or if not previously used are set to the values described in the initializing procedure), the following calls are made.

```
CALL ERASE
CALL CONV3D(N,ARRAY,NUSED)
CALL HDNSRF
CALL ALTER
CALL CMPUT
```

The call to ERASE need only be made after a previous view has been developed. Calling ERASE before developing the first view will not degrade the result however. This subroutine resets flags in the structure. It will also reconnect the lines to their proper points if the subroutine ALTER was used to modify the structure.

The subroutine CONV3D converts the three-dimensional points to the two-dimensional points in the picture plane. The arguments have been previously defined. Additionally CONV3D causes the position of the observer to be calculated in terms of the object's coordinate system. This information is used during hidden-line removal.

Some surfaces of an object will be hidden because they face away from the observer. Subroutine HDNSRF identifies those faces and eliminates them from further consideration for the view being developed.

The call to ALTER only need be made when a portion of the picture is behind the observer. This subroutine cuts off the lines passing through the observer's viewing plane. If it is known in advance that this condition will not exist, ALTER need not be used. ALTER modifies the line structure for those lines passing through the observer's viewing plane.

The final call to finish developing the picture is to CMPUT. Each on-view line is traversed and those visible portions are specified to be drawn. Those portions not visible are suppressed.

The form of the picture is now an array of NLINES lines with starting points in XSTART and YSTART and end points in XEND and YEND. Any type of display device may now be used for output using only this list of lines.

H. CONVENIENCE SUBROUTINES

Several helpful subroutines are included to aid the user. These subroutines are not necessary to obtain a picture, but are helpful in using the program. The calls to these subroutines are listed for reference.

```
CALL DATSTR
CALL LABEL
CALL DRAWVP(NLINES,XSTART,YSTART,XEND,YEND)
CALL DRAWIT(NLINES,XSTART,YSTRAT,XEND,YEND)
```

1. Subroutine DATSTR

This subroutine will cause a printout of the data stored in ARRAY(N,13). The single-subscript arrays LSTFAC and LSTBFC will also be printed. The starting and end points of the lines in the picture plane are also printed (visible lines).

2. Subroutine LABEL

This subroutine will printout the angles and positions of the object and the observer, and also the offsets, focus, and scale. This subprogram is convenient for data taking.

3. Subroutine DRAWVP

This subprogram gives a line printer drawing of the picture developed and is useful in program development. Lines are drawn by six different synbols--i.e., the first six lines are each printed with a different symbol, with a maximum of ten per line. Then the symbols are repeated for every six lines. DRAWVP is dimensioned for 100 lines. If more lines are drawn, the dimensions of DRAW and JXY in DRAWVP must be increased, but the picture will become quite confused.

This subroutine requires another subroutine to be furnished. The required subroutine is VPLOT and is available in the primary library of the Naval Postgraduate School SDS-9300 computer. When using the IBM 360-67 computer, the subroutine must be furnished. VPLOT was developed for the SDS-9300 computer at the Electrical Engineering Computer Laboratory by Professor R. C. Johnson. Changes in the symbols used in plotting must be made in VPLOT.

4. Subroutine DRAWIT

For a hard copy drawn by the CALCOMP plotter, the subroutine DRAWIT is convenient. This subroutine utilizes the local IBM 360-67 NAVPGSCOL plotting package (Ref. 35).

This subroutine should be developed when needed to fit a specific application. The subroutine DRAWIT is used as an example to show how the plotting package can be utilized in conjunction with the three-dimensional program. The title and identification specifically are user functions. Before using this subroutine, the user should consult the referenced technical note.

An alternative to this subroutine is the IBM 360-67 DRAW routine which also utilizes the plotting package. DRAW is a much more general program and thus will be more expensive in computer storage and computing time.

I. INTERPRETING THE STRUCTURED DATA

Certain problems will occur if the data is not correctly entered. The fastest way to check the data structure is to call DATSTR. Studying the printout may reveal where the problem lies.

The program assumes that every line is the intersection of two planes. The two planes involved are stored in column 11 of the array. This requires that all lines must be entered at least once in each direction. The same points used to enter a line in one direction must be used to enter the line in the reverse direction. If a line is entered as two segments in one direction, it is necessary to have it entered as two segments in the opposite direction. A line can be drawn on a plane by entering the line first in one direction and then entering it in the reverse direction. Figure 31 illustrates these two special cases.

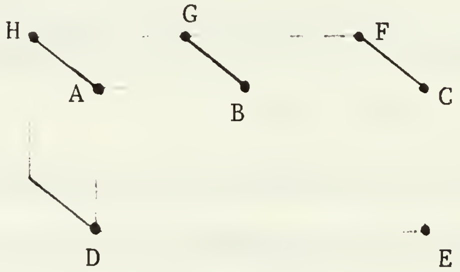


FIGURE 31

ENTERING LINES

Let surface 1 be ADECB and surface 2 be ABCFGH. The surfaces intersect along ABC. Line BG is a line to be drawn that lies entirely on surface 2. Surface 2 can be entered by entering the coordinates for the points in the order A B G B C F G H. The line BG is then entered as part of surface 2 first in the direction BG and then in the direction GB. When entering surface 1, the order can be A D E C B. Line CA must be entered as two segments (CB and BA) since the surface that it intersects with is entered as AB and BC.

In the data structure, column 9 is the line list which lists the end points of each line. Column 11 is a line list which lists the two surfaces which intersect to form the line. Column 10 is a continuation of column 9 and column 12 is a continuation of column 11. At the end of the line lists one space is skipped and the remaining space is used for scratch storage during computation. If both integers are not filled in column 11, the line was entered in only one direction. The line must be entered in both directions.

The minus signs are used to mark the various segments. Hidden lines are marked with a minus in the first integer of column 9. A line that has been examined over its length for hidden parts is marked with a minus in the second integer. A minus in the first integer of column 11 indicates the line is an intersection of two planes forming a concave dihedral.

Column 13 refers to the associated points. The first integer specifies the number of surfaces obstructing a point and the second integer is a pointer to the single-subscript array LSTBFC which identifies these surfaces.

Some points will have zero for both integers. These points were hidden and thus never entered into the computation. Those points which have a zero for the first integer but a number for the second integer have no surfaces blocking them from view. The current pointer is always entered whenever the nature (number of faces blocking the point) is found as an indication that the nature is now known.

Column 8 is a list of the surfaces. The first integer specifies the number of points in the surface. The second integer is a pointer to LSTFAC array which contains each point in the order entered. A minus in front of the first integer indicates that surface is oriented away from the observer such that it is not on view.

Columns 1 - 3 are the original coordinates entered. Column 4 is a depth column for the points in the current view. Columns 5 and 6 are picture plane coordinates of the points. Column 5 is the horizontal component and column 6 is the vertical component. The first integer in column 7 when set to 1 indicates the point is an end point of at least one concave dihedral. The second integer in column 7 is not used.

If difficulties cannot be resolved by examination of the structured data alone, reference between the algorithm explained in chapter VI and the data structure should reveal the nature of the difficulty.

```
   PARAMETERS FOR THIS VIEW

OBJECT POSITION (1)                        -2.5
OBJECT POSITION (2)                        -5.0
OBJECT POSITION (3)                        -2.5
OBJECT ANGLE ALPHA                          0.0
OBJECT ANGLE BETA                           0.0
OBJECT ANGLE GAMMA                          0.0
OBSERVERS POSITION (1)                     40.0
OBSERVERS POSITION (2)                      0.0
OBSERVERS POSITION (3)                     40.0
OBSERVERS ANGLE PSI                         0.0
OBSERVERS ANGLE THETA                     -35.0
OBSERVERS ANGLE PHI                         0.0
FOCAL LENGTH                                1.5
SCALE                                       0.5
HORIZONTAL OFFSET                           0.0
VERTICAL OFFSET                             0.0
```

TABLE I

PARAMETERS AND STRUCTURED DATA FOR L-BLOCK (1)

```
COLUMN       1            2           3            4             5             6       7       13
IDENT     THREE-DIMENSIONAL COORDINATES          DEPTH      PICTURE PLANE COORD CCV/CVX NATURE
   1        0.0        10.0000       0.0        -59.1910      0.2472       -0.5159      0       0  1
   2        0.0        10.0000       5.0000     -56.3231      0.2594       -0.3290      0       0  1
   3        0.0         0.0          5.0000     -56.3231     -0.2594       -0.3290      0       0  1
   4        0.0         0.0          0.0        -59.1910     -0.2472       -0.5159      0       0  1
   5       -5.0000     10.0000       0.0        -63.2867      0.2315       -0.3505      0       0  1
   6       -5.0000     10.0000       5.0000     -60.4188      0.2423       -0.1683      0       0  1
   7       -5.0000      3.0000       0.0        -63.2867     -0.0926       -0.3505      1       1  1
   8      -16.0000      3.0000       0.0        -72.2974     -0.0813       -0.0512      0       0  2
   9      -16.0000      3.0000       5.0000     -69.4295     -0.0846        0.1200      0       0  2
  10       -5.0000      3.0000       5.0000     -60.4188     -0.0969       -0.1683      1       0  2
  11      -16.0000      0.0          0.0        -72.2974     -0.2033       -0.0512      0       0  1
  12      -16.0000      0.0          5.0000     -69.4295     -0.2115        0.1200      0       0  1
```

```
COLUMN    8              9          10              11         12
IDENT   FACES          LINE END POINTS         INTERSECTING SURFACES
  1       4    1        1   -2         6 -10      1    2         8   4
  2      -4    5        2   -3        -8  11      1    8         5   7
  3       4    9        3   -4       -11  12      1    6         5   6
  4      -4   13        1   -4         9 -12      7    1         8   5
  5      -4   17       -1    5        -4  11      2    7         7   6
  6      -4   21       -5    6         3 -12      2    4         6   8
  7      -6   25        2   -6         0   0      8    2         0   0
  8       6   31        7   -8        -2   0      3    7         1   1
  9       0    0        8   -9       -13   0      3    5         8   4
 10       0    0        9  -10         0   0      3    8         3   2
 11       0    0       -7   10         0   0     -4    3         0   0
 12       0    0       -5    7         0   0      4    7         0   0
```

```
                              LSTFAC

  1      2      3      4      1          5      6      2      7      8
  9     10      5      7     10          6      8     11     12      9
 11      4      3     12      4         11      8      7      5      1
  3      2      6     10      9         12

                              LSTBFC

  1
```

```
                         VISIBLE LINES

    XSTART       YSTART                XEND       YEND
  0.247154    -0.515904            0.259412   -0.328994
  0.247154    -0.515904           -0.247154   -0.515904
  0.259412    -0.328994           -0.259412   -0.328994
  0.259412    -0.328994            0.242253   -0.168281
 -0.259412    -0.328994           -0.247154   -0.515904
 -0.259412    -0.328994           -0.211478    0.119954
 -0.085727    -0.168279           -0.081304   -0.051208
 -0.081304    -0.051208           -0.084591    0.119954
 -0.084591     0.119954           -0.096901   -0.168281
 -0.084591     0.119954           -0.211478    0.119954
  0.242253    -0.168281           -0.096901   -0.168281
```

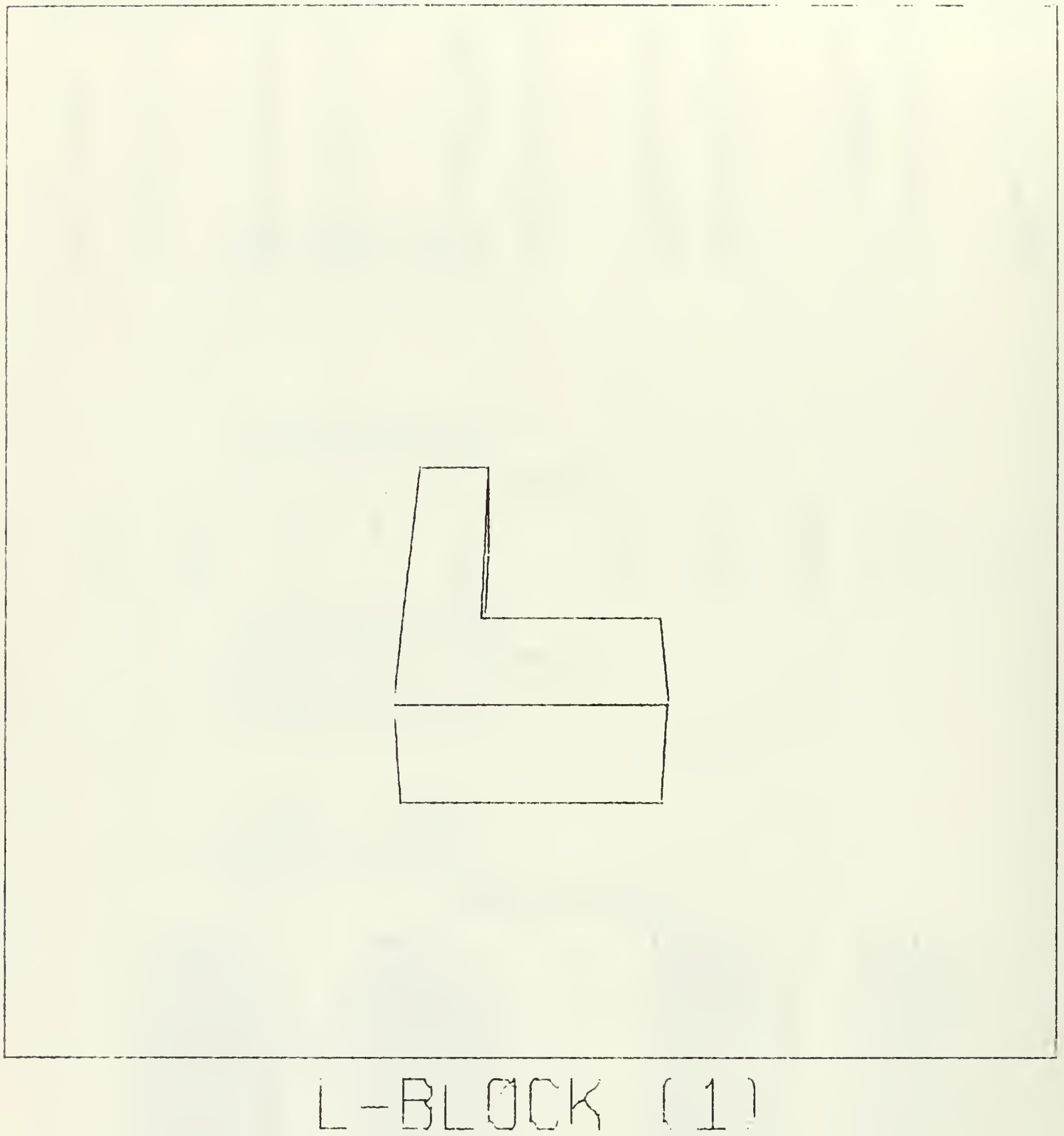


FIGURE 32

```
   PARAMETERS FOR THIS VIEW

OBJECT POSITION (1)                        -2.5
OBJECT POSITION (2)                        -5.0
OBJECT POSITION (3)                        -2.5
OBJECT ANGLE ALPHA                          0.0
OBJECT ANGLE BETA                           0.0
OBJECT ANGLE GAMMA                          0.0
OBSERVERS POSITION (1)                    -50.0
OBSERVERS POSITION (2)                     30.0
OBSERVERS POSITION (3)                     60.0
OBSERVERS ANGLE PSI                       149.0
OBSERVERS ANGLE THETA                     -46.5
OBSERVERS ANGLE PHI                         0.0
FOCAL LENGTH                                1.5
SCALE                                       0.5
HORIZONTAL OFFSET                           0.0
VERTICAL OFFSET                             0.0
```

TABLE II

PARAMETERS AND STRUCTURED DATA FOR L-BLOCK (2)

```
COLUMN         1              2              3              4              5                6         7       13
IDENT    THREE-DIMENSICNAL COORDINATES                    DEPTH        PICTURE PLANE COORD  CCV/CVX NATURE
  1          0.0         10.0000         0.0          -82.2258      -0.1087         -0.1486       0       0  1
  2          0.0         10.0000         5.0000       -78.5989      -0.1137         -0.0265       0       0  1
  3          0.0          0.0            5.0000       -82.1442       0.1986          0.1087       0       0  1
  4          0.0          0.0            0.0          -85.7717       0.1903         -0.0142       0       0  0
  5         -5.0000      10.0000         0.0          -79.2756      -0.0171         -0.2695       0       0  1
  6         -5.0000      10.0000         5.0000       -75.6487      -0.0179         -0.1484       0       0  1
  7         -5.0000       3.0000         0.0          -81.7573       0.1996         -0.1673       1       0  1
  8        -16.0000       3.0000         0.0          -75.2669       0.4379         -0.4487       0       0  1
  9        -16.0000       3.0000         5.0000       -71.6400       0.4596         -0.3298       0       0  1
 10         -5.0000       3.0000         5.0000       -78.1304       0.2087         -0.0452       1       0  1
 11        -16.0000       0.0            0.0          -76.3305       0.5311         -0.3994       0       0  1
 12        -16.0000       0.0            5.0000       -72.7036       0.5570         -0.2797       0       0  1
```

```
COLUMN     8                 9          10          11           12
IDENT   FACES            LINE END POINTS       INTERSECTING SURFACES
  1       -4    1         1   -2        6 -10       1    2       8    4
  2        4    5         2   -3        8 -11       1    8       5    7
  3        4    9        -3    4       11 -12       1    6       5    6
  4        4   13        -1    4        9 -12       7    1       8    5
  5        4   17         1   -5       -4  11       2    7       7    6
  6       -4   21         5   -6        3 -12       2    4       6    8
  7       -6   25         2   -6        0   0       8    2       0    0
  8        6   31         7   -8       -2   0       3    7       2    2
  9        0    0         8   -9      -13   0       3    5       8    2
 10        0    0         9  -10        0   0       3    8       3    2
 11        0    0         7  -10        0   0      -4    3       0    0
 12        0    0         5   -7        0   0       4    7       0    0
```

```
                                LSTFAC

     1    2    3    4    1          5    6    2    7    8
     9   10    5    7   10          6    8   11   12    9
    11    4    3   12    4         11    8    7    5    1
     3    2    6   10    9         12

                                LSTBFC
NULL  SET
```

```
                          VISIBLE LINES

       XSTART       YSTART               XEND       YEND
   -0.108744    -0.148635           -0.113668  -0.026458
   -0.108744    -0.148635           -0.017074  -0.269526
   -0.113668    -0.026458            0.198584   0.108657
   -0.113668    -0.026458           -0.017877  -0.148361
   -0.017074    -0.269526           -0.017877  -0.148361
   -0.017074    -0.269526            0.199639  -0.167261
    0.199639    -0.167261            0.437917  -0.448684
    0.199639    -0.167261            0.208732  -0.045214
    0.437917    -0.448684            0.459633  -0.329762
    0.437917    -0.448684            0.531053  -0.399352
    0.459633    -0.329762            0.208732  -0.045214
    0.459633    -0.329762            0.557009  -0.279723
   -0.017877    -0.148361            0.208732  -0.045214
    0.531053    -0.399352            0.557009  -0.279723
    0.198584     0.108657            0.557009  -0.279723
```

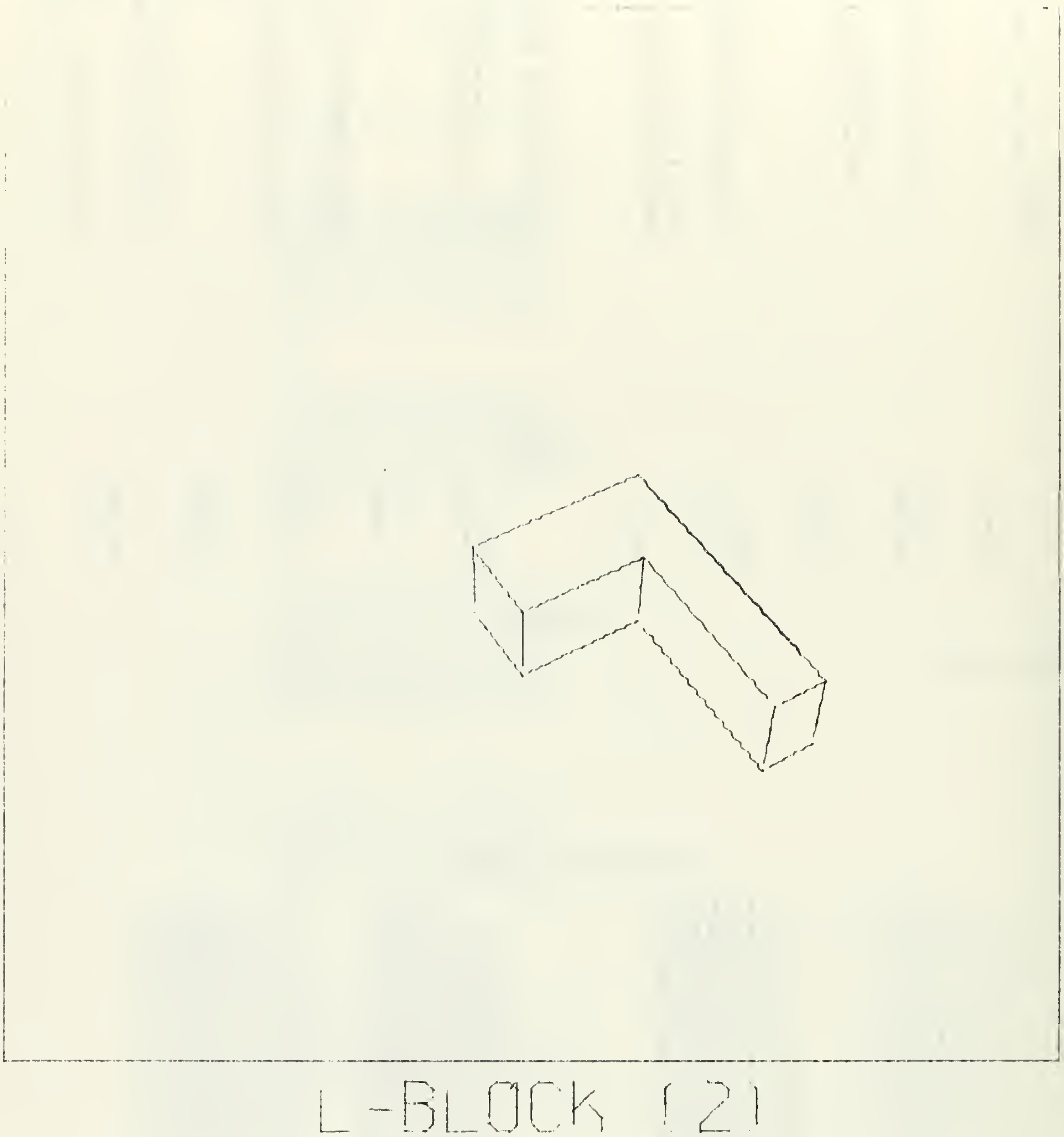


FIGURE 33

```
   PARAMETERS FOR THIS VIEW

OBJECT POSITION (1)                         -2.5
OBJECT POSITION (2)                         -5.0
OBJECT POSITION (3)                         -2.5
OBJECT ANGLE ALPHA                           0.0
OBJECT ANGLE BETA                            0.0
OBJECT ANGLE GAMMA                           0.0
OBSERVERS POSITION (1)                     -50.0
OBSERVERS POSITION (2)                     -50.0
OBSERVERS POSITION (3)                      50.0
OBSERVERS ANGLE PSI                        225.0
OBSERVERS ANGLE THETA                      -35.0
OBSERVERS ANGLE PHI                          0.0
FOCAL LENGTH                                 1.5
SCALE                                        0.5
HORIZONTAL OFFSET                            0.0
VERTICAL OFFSET                              0.0
```

TABLE III

PARAMETERS AND STRUCTURED DATA FOR L-BLOCK (3)

```
COLUMN        1            2             3            4              5           6        7       13
IDENT    THREE-DIMENSIONAL COORDINATES             DEPTH      PICTURE PLANE COORD  CCV/CVX  NATURE
   1        0.0        10.0000        0.0        -89.4836      -0.1748     -0.0473     0      0  1
   2        0.0        10.0000        5.0000     -86.6157      -0.1805      0.0906     0      0  1
   3        0.0         0.0           5.0000     -80.8234       0.0644     -0.0508     0      0  1
   4        0.0         0.0           0.0        -83.6913       0.0623     -0.1933     0      0  1
   5       -5.0000     10.0000        0.0        -86.5875      -0.3010     -0.1179     0      0  1
   6       -5.0000     10.0000        5.0000     -83.7196      -0.3111      0.0223     0      0  1
   7       -5.0000      3.0000        0.0        -82.5329      -0.1388     -0.2249     1      1  1
   8      -16.0000      3.0000        0.0        -76.1614      -0.4507     -0.4157     0      0  2
   9      -16.0000      3.0000        5.0000     -73.2935      -0.4680     -0.2674     0      0  2
  10       -5.0000      3.0000        5.0000     -79.6650      -0.1437     -0.0815     1      0  2
  11      -16.0000      0.0           0.0        -74.4237      -0.3772     -0.4733     0      0  2
  12      -16.0000      0.0           5.0000     -71.5558      -0.3920     -0.3237     0      0  1
```

```
COLUMN     8               9            10             11           12
IDENT   FACES         LINE END POINTS          INTERSECTING SURFACES
  1       -4    1     -1    2        6 -10         1    2         8    4
  2       -4    5      2   -3        8 -11         1    8         5    7
  3       -4    9      3   -4       11 -12         1    6         5    6
  4        4   13     -1    4        9 -12         7    1         8    5
  5        4   17     -1    5        4 -11         2    7         7    6
  6        4   21      5   -6        3 -12         2    4         6    8
  7       -6   25      2   -6        0   0         8    2         0    0
  8        6   31     -7    8      -10   0         3    7         5    2
  9        0    0      8   -9      -18   0         3    5         8    2
 10        0    0      9  -10        0   0         3    8         3    2
 11        0    0     -7   10        0   0        -4    3         0    0
 12        0    0      5   -7        0   0         4    7         0    0
```

```
                          LSTFAC

    1     2     3     4     1         5     6     2     7     8
    9    10     5     7    10         6     8    11    12     9
   11     4     3    12     4        11     8     7     5     1
    3     2     6    10     9        12

                          LSTBFC

    6
```

```
                        VISIBLE LINES

      XSTART         YSTART              XEND         YEND
   -0.180539       0.090642          0.064438    -0.050782
   -0.180539       0.090642         -0.311137     0.022334
    0.064438      -0.050782          0.062269    -0.193304
    0.064438      -0.050782         -0.391983    -0.323702
   -0.301007      -0.117883         -0.311137     0.022334
   -0.301007      -0.117883         -0.257406    -0.146661
   -0.450681      -0.415719         -0.467962    -0.267376
   -0.450681      -0.415719         -0.377176    -0.473311
   -0.467962      -0.267376         -0.143730    -0.081488
   -0.467962      -0.267376         -0.391983    -0.323702
   -0.311137       0.022334         -0.143730    -0.081488
   -0.377176      -0.473311         -0.391983    -0.323702
    0.062269      -0.193304         -0.377176    -0.473311
```

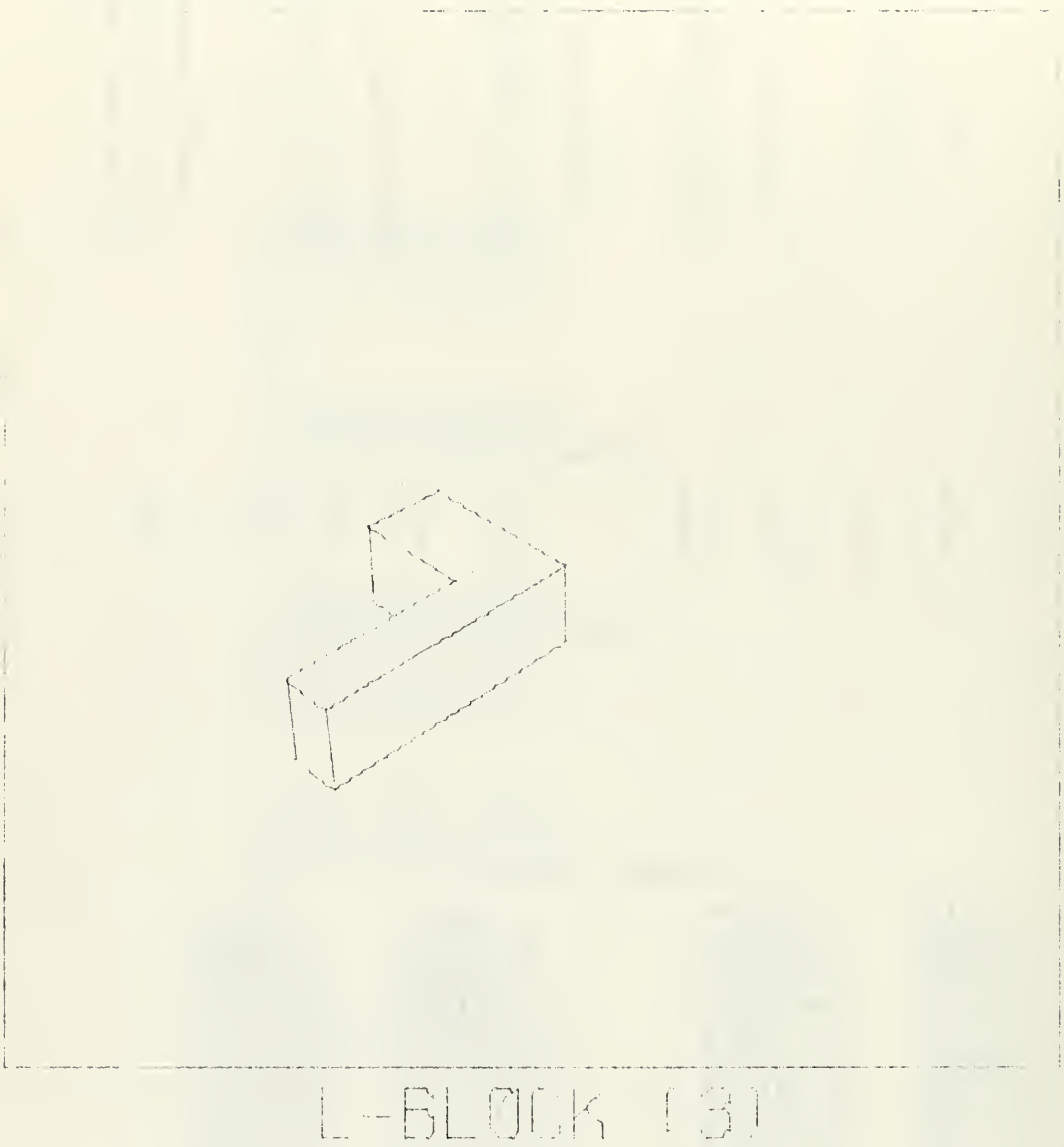


FIGURE 34

```
   PARAMETERS FOR THIS VIEW

OBJECT POSITION (1)                       2000.0
OBJECT POSITION (2)                          0.0
OBJECT POSITION (3)                          0.0
OBJECT ANGLE ALPHA                          60.0
OBJECT ANGLE BETA                            0.0
OBJECT ANGLE GAMMA                           0.0
OBSERVERS POSITION (1)                       0.0
OBSERVERS POSITION (2)                       0.0
OBSERVERS POSITION (3)                     500.0
OBSERVERS ANGLE PSI                        190.0
OBSERVERS ANGLE THETA                       -5.0
OBSERVERS ANGLE PHI                          0.0
FOCAL LENGTH                                 1.5
SCALE                                        0.5
HORIZONTAL OFFSET                            0.0
VERTICAL OFFSET                              0.0
```

TABLE IV

PARAMETERS AND STRUCTURED DATA FOR CARRIER (1)

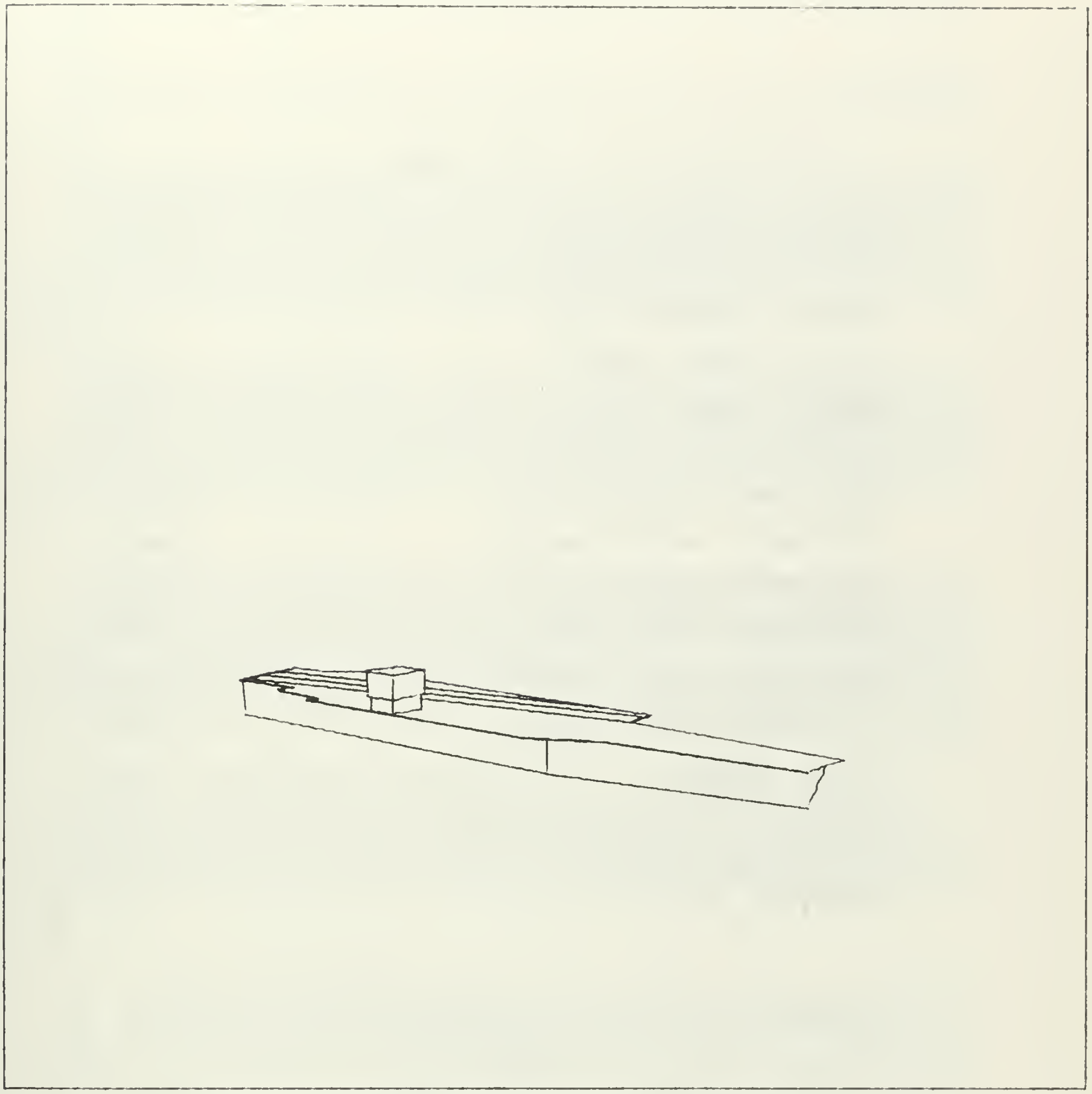
```
COLUMN         1                2             3            4             5              6          7         13
IDENT     THREE-DIMENSIONAL COORDINATES                  DEPTH      PICTURE PLANE COORD  CCV/CVX  NATURE
   1         0.0          45.6000        57.0000  -1965.9297       0.4849        -0.4158       0        0  1
   2         0.0           0.0           57.0000  -2000.7283       0.5204        -0.4040       0        0  1
   3         0.0         -45.6000        57.0000  -2035.5269       0.5547        -0.3926       1        0  1
   4       353.3999      -64.0000        57.0000  -2275.8660       0.1551        -0.3235       1        0  1
   5       365.0000     -131.0000        57.0000  -2334.4236       0.1951        -0.3088       0        0  1
   6       955.0000     -131.0000        57.0000  -2712.2268      -0.3317        -0.2293       0        0  1
   7      1009.0000      -79.8000        57.0000  -2707.7334      -0.4145        -0.2301       1        0  1
   8      1068.0000      -68.5000        57.0000  -2736.8904      -0.4676        -0.2249       0        0  1
   9      1046.0000       62.7000        57.0000  -2622.6807      -0.5651        -0.2461       0        0  1
  10      1052.0000       45.6000        57.0000  -2639.5723      -0.5542        -0.2428       0        0  1
  11       353.0000      -64.0000        57.0000  -2275.6099       0.1555        -0.3236       0        0  1
  12      1058.5000        5.7000        57.0000  -2674.1831      -0.5238        -0.2363       0        0  1
  13       358.5000     -103.9000        57.0000  -2309.5806       0.1810        -0.3150       0        0  1
  14      1065.0000      -33.5000        57.0000  -2708.2600      -0.4949        -0.2300       0        0  1
  15       450.5000     -131.0000        57.0000  -2389.1729       0.1084        -0.2957       0        0  1
  16       930.0000       50.2000        57.0000  -2557.9397      -0.4658        -0.2589       1        0  1
  17       930.0000       85.6000        57.0000  -2530.9250      -0.4977        -0.2645       0        0  1
  18       828.0000       90.0000        57.0000  -2462.2522      -0.4199        -0.2792       1        0  1
  19       828.0000      114.0000        57.0000  -2443.9370      -0.4419        -0.2832       0        0  1
  20       647.0000      114.0000        57.0000  -2328.0347      -0.2854        -0.3104       0        0  1
  21       647.0000      110.0000        57.0000  -2331.0872      -0.2817        -0.3096       1        0  1
  22       694.0000      110.0000        57.0000  -2361.1833      -0.3238        -0.3023       1        0  1
  23       694.0000       52.5000        57.0000  -2405.0632      -0.2718        -0.2920       1        0  1
  24       647.0000       52.5000        57.0000  -2374.9670      -0.2298        -0.2991       1        0  1
  25       394.0000      114.0000        57.0000  -2166.0276      -0.0385        -0.3532       0        0  1
  26       302.0000       63.0000        57.0000  -2146.0354       0.1054        -0.3589       1        0  1
  27       694.0000      110.0000        79.8000  -2359.1963      -0.3241        -0.2737       1        1  2
  28       694.0000       52.5000        79.8000  -2403.0762      -0.2721        -0.2640       1        0  2
  29       698.0000       48.5000        79.8000  -2408.6899      -0.2721        -0.2627       0        0  2
  30       698.0000      114.0000        79.8000  -2358.7051      -0.3313        -0.2738       0        0  3
  31       698.0000      114.0000       114.0000  -2355.7244      -0.3317        -0.2308       0        0  3
  32       698.0000       48.5000       114.0000  -2405.7092      -0.2724        -0.2206       0        0  3
  33       643.0000       48.5000        79.8000  -2373.4709      -0.2229        -0.2705       0        0  3
  34       643.0000       48.5000       114.0000  -2370.4902      -0.2231        -0.2278       0        0  3
  35       647.0000       52.5000        79.8000  -2372.9800      -0.2300        -0.2706       1        1  3
  36       643.0000      114.0000       114.0000  -2320.5054      -0.2823        -0.2383       0        0  3
  37       643.0000      114.0000        79.8000  -2323.4861      -0.2820        -0.2820       0        0  3
  38       647.0000      110.0000        79.8000  -2329.1001      -0.2819        -0.2807       1        1  1
  39        25.0000        0.0            0.0     -2021.7048       0.4866        -0.4820       0        0  4
  40       394.0000       66.5000         0.0     -2207.2441       0.0037        -0.4194       0        0  4
```

```
COLUMN      1             2              3            4               5              6         7        13
IDENT    THREE-DIMENSIONAL COORDINATES              DEPTH        PICTURE PLANE COORD CCV/CVX NATURE
 41      1040.0000      61.0000       0.0        -2625.1040      -0.5580      -0.3109      0       0  5
 42      1040.0000     -67.0000       0.0        -2722.7842      -0.4474      -0.2904      0       0  0
 43       394.0000     -66.5000       0.0        -2308.7400       0.1146      -0.3895      0       0  0
 44      1046.0000     -67.0000      56.0000     -2721.7454      -0.4527      -0.2287      1       0  0
 45      1046.0000      61.0000      56.0000     -2624.0652      -0.5635      -0.2469      1       1  6
 46       394.0000     -66.5000      56.0000     -2303.8591       0.1148      -0.3177      1       0  0
 47         0.0          0.0         56.0000     -2000.8154       0.5204      -0.4055      1       1  4
 48       394.0000      66.5000      56.0000     -2202.3633       0.0037      -0.3444      1       1  5
 49         0.0         45.6000      56.0000     -1966.0168       0.4849      -0.4173      0       0  1
 50       302.0000      63.0000      56.0000     -2146.1226       0.1054      -0.3603      1       0  7
 51       394.0000     114.0000      56.0000     -2166.1147      -0.0385      -0.3546      0       0  2
 52       828.0000     114.0000      56.0000     -2444.0242      -0.4419      -0.2844      0       0  1
 53       828.0000      90.0000      56.0000     -2462.3394      -0.4199      -0.2804      1       1  7
 54       930.0000      85.6000      56.0000     -2531.0122      -0.4977      -0.2657      0       0  1
 55       930.0000      50.2000      56.0000     -2558.0269      -0.4658      -0.2601      1       1  8
 56      1046.0000      62.7000      56.0000     -2622.7678      -0.5650      -0.2472      0       0  1
 57      1068.0000     -68.5000      56.0000     -2736.9775      -0.4676      -0.2259      0       0  0
 58      1009.0000     -79.8000      56.0000     -2707.8206      -0.4145      -0.2312      1       0  0
 59       955.0000    -131.0000      56.0000     -2712.3140      -0.3317      -0.2304      0       0  0
 60       365.0000    -131.0000      56.0000     -2334.5107       0.1951      -0.3101      0       0  1
 61       353.3999     -64.0000      56.0000     -2275.9531       0.1551      -0.3248      1       1  9
 62         0.0        -45.6000      56.0000     -2035.6140       0.5546      -0.3941      0       0  1
 63         0.0          0.0          0.0            0.0           0.0          0.0        0       0  0
 64         0.0          0.0          0.0            0.0           0.0          0.0        0       0  0
 65         0.0          0.0          0.0            0.0           0.0          0.0        0       0  0
 66         0.0          0.0          0.0            0.0           0.0          0.0        0       0  0
 67         0.0          0.0          0.0            0.0           0.0          0.0        0       0  0
 68         0.0          0.0          0.0            0.0           0.0          0.0        0       0  0
 69         0.0          0.0          0.0            0.0           0.0          0.0        0       0  0
 70         0.0          0.0          0.0            0.0           0.0          0.0        0       0  0
 71         0.0          0.0          0.0            0.0           0.0          0.0        0       0  0
 72         0.0          0.0          0.0            0.0           0.0          0.0        0       0  0
 73         0.0          0.0          0.0            0.0           0.0          0.0        0       0  0
 74         0.0          0.0          0.0            0.0           0.0          0.0        0       0  0
 75         0.0          0.0          0.0            0.0           0.0          0.0        0       0  0
 76         0.0          0.0          0.0            0.0           0.0          0.0        0       0  0
 77         0.0          0.0          0.0            0.0           0.0          0.0        0       0  0
 78         0.0          0.0          0.0            0.0           0.0          0.0        0       0  0
 79         0.0          0.0          0.0            0.0           0.0          0.0        0       0  0
 80         0.0          0.0          0.0            0.0           0.0          0.0        0       0  0
```

```
COLUMN      8                9               10                11              12
IDENT   FACES           LINE END POINTS            INTERSECTING SURFACES
   1     36    1           1   -2        1  -49           1   32          32   19
   2     -4   37           2   -3       25  -51           1   32          20   21
   3     -4   41           3   -4       19  -52           1   31          21   22
   4     -4   45           4   -5      -18   53           1   30         -22   23
   5     -4   49           5   -6       17  -54           1   29          23   24
   6      4   53           6   -7      -16   55           1   28         -24   25
   7      4   57           7   -8        9  -56           1   27          25   26
   8      4   61           8   -9       -8   57           1   26          26   27
   9      4   65           9  -10       -7   58           1    1         -27   28
  10      4   69          10  -11       -6   59           1    1          28   29
  11     -5   73          10  -12        5  -60           1    1          29   30
  12     -4   78          12  -13       -4   61           1    1         -30   31
  13     -4   82          12  -14        3  -62           1    1          31   32
  14     -4   86          14  -15        0    0           1    1           0    0
  15      4   90           8  -14       -3    0           1    1           1    1
  16      4   94           9  -16      -66    0           1   25          25    2
  17    -20   98          16  -17      -42    0           1   24          23    2
  18    -10  118          17  -18      -43    0           1   23          32    1
  19      4  128          18  -19      -44    0           1   22           0    0
  20      4  132          19  -20        0    0           1   21           0    0
  21      5  136          20  -21        0    0           1    1           0    0
  22     -4  141          21  -22        0    0          -1    9           0    0
  23      4  145         -22   23        0    0          -1    2           0    0
  24     -4  149         -23   24        0    0          -1    5           0    0
  25      4  153          21  -24        0    0          -7    1           0    0
  26     -4  157          20  -25        0    0           1   21           0    0
  27     -4  161          25  -26        0    0           1   20           0    0
  28     -4  165           1  -26        0    0          19    1           0    0
  29     -4  169          22  -27        0    0           2    9           0    0
  30      4  173         -27   28        0    0          -2   18           0    0
  31     -4  177         -23   28        0    0           5    2           0    0
  32      6  181         -29   30        0    0           3   18           0    0
  33      0    0          30  -31        0    0           3    8           0    0
  34      0    0          31  -32        0    0           3   10           0    0
  35      0    0         -29   32        0    0           4    3           0    0
  36      0    0         -29   33        0    0          18    4           0    0
  37      0    0          32  -34        0    0           4   10           0    0
  38      0    0          33  -34        0    0           6    4           0    0
  39      0    0         -28   35        0    0          -5   18           0    0
  40      0    0          24  -35        0    0           7    5           0    0
  41      0    0          34  -36        0    0           6   10           0    0
  42      0    0          36  -37        0    0           6    8           0    0
  43      0    0          33  -37        0    0          18    6           0    0
  44      0    0         -35   38        0    0          -7   18           0    0
  45      0    0          21  -38        0    0           9    7           0    0
  46      0    0          30  -37        0    0           8   18           0    0
  47      0    0          31  -36        0    0          10    8           0    0
  48      0    0         -27   38        0    0         -18    9           0    0
  49      0    0          39  -40        0    0          11   15           0    0
  50      0    0          40  -41        0    0          11   16           0    0
  51      0    0         -41   42        0    0          11   12           0    0
  52      0    0         -42   43        0    0          11   13           0    0
  53      0    0         -39   43        0    0          14   11           0    0
  54      0    0         -42   44        0    0          13   12           0    0
  55      0    0          41  -45        0    0          12   16           0    0
  56      0    0         -44   45        0    0         -17   12           0    0
  57      0    0         -43   46        0    0          14   13           0    0
  58      0    0         -44   46        0    0         -13   17           0    0
  59      0    0          39  -47        0    0          15   14           0    0
  60      0    0         -46   47        0    0         -14   17           0    0
  61      0    0         -47   48        0    0         -15   17           0    0
  62      0    0          40  -48        0    0          16   15           0    0
  63      0    0         -45   48        0    0         -17   16           0    0
  64      0    0          47  -49        0    0          17   32           0    0
```

```
COLUMN    8           9      10          11         12
IDENT  FACES       LINE END POINTS     INTERSECTING SURFACES
 65     0   0      49 -50     0   0     17  19      0   0
 66     0   0      50 -51     0   0     17  20      0   0
 67     0   0      51 -52     0   0     17  21      0   0
 68     0   0     -52  53     0   0     17  22      0   0
 69     0   0      53 -54     0   0     17  23      0   0
 70     0   0     -54  55     0   0     17  24      0   0
 71     0   0      55 -56     0   0     17  25      0   0
 72     0   0     -56  57     0   0     17  26      0   0
 73     0   0     -57  58     0   0     17  27      0   0
 74     0   0     -58  59     0   0     17  28      0   0
 75     0   0     -59  60     0   0     17  29      0   0
 76     0   0      60 -61     0   0     17  30      0   0
 77     0   0     -61  62     0   0     17  31      0   0
 78     0   0      47 -62     0   0     32  17      0   0
 79     0   0     -37  38     0   0     18  18      0   0
 80     0   0      26 -50     0   0    -19  20      0   0




                                LSTFAC

    1     2     3     4     5          6     7     8     9    10
   11    10    12    13    12         14    15    14     8    14
   12    10     9    16    17         18    19    20    21    22
   23    24    21    20    25         26    23    22    27    28
   29    30    31    32    33         29    32    34    24    23

   28    35    33    34    36         37    24    35    38    21
   30    37    36    31    22         21    38    27    34    32
   31    36    39    40    41         42    43    44    42    41
   45    46    43    42    44         47    39    43    46    39
   47    48    40    40    48         45    41    47    46    44

   45    48    47    49    50         51    52    53    54    55
   56    57    58    59    60         61    62    38    35    28
   27    38    37    30    29         33    37    26    50    49
    1    25    51    50    26         19    52    51    25    20
   18    53    52    19    17         54    53    18    16    55

   54    17     9    56    55         16     8    57    56     9
    7    58    57     8     6         59    58     7     5    60
   59     6     4    61    60          5     3    62    61     4
    3     2     1    49    47         62

                                LSTBFC

    8     8     6    32     1         25     1     1     1
```

```
                     VISIBLE LINES

   XSTART        YSTART                    XEND          YEND
 0.484892     -0.415814                0.520382     -0.404026
 0.484892     -0.415814                0.105424     -0.358930
 0.484892     -0.415814                0.484870     -0.417315
 0.520382     -0.404026                0.554660     -0.392640
 0.554660     -0.392640                0.155084     -0.323504
 0.554660     -0.392640                0.554636     -0.394091
 0.155084     -0.323504                0.195094     -0.308815
 0.195094     -0.308815               -0.223027     -0.245690
-0.326935     -0.230002               -0.331709     -0.229282
 0.195094     -0.308815                0.195087     -0.310083
-0.331709     -0.229282               -0.414508     -0.230097
-0.414508     -0.230097               -0.467567     -0.224853
-0.467567     -0.224853               -0.565062     -0.246062
-0.467567     -0.224853               -0.494870     -0.230002
-0.565062     -0.246062               -0.554183     -0.242809
-0.565062     -0.246062               -0.465780     -0.258925
-0.565062     -0.246062               -0.565062     -0.246941
-0.565062     -0.247193               -0.565043     -0.247193
-0.554183     -0.242809               -0.331383     -0.268164
-0.229969     -0.279704                0.155505     -0.323570
-0.554183     -0.242809               -0.523841     -0.236274
-0.523841     -0.236274               -0.331484     -0.257745
-0.222869     -0.269870                0.181043     -0.314956
-0.523841     -0.236274               -0.494870     -0.230002
-0.494870     -0.230002               -0.331581     -0.247792
-0.222936     -0.259629                0.108436     -0.295732
-0.465780     -0.258925               -0.497705     -0.264487
-0.497705     -0.264487               -0.419879     -0.279175
-0.497705     -0.264487               -0.497688     -0.265658
-0.419879     -0.279175               -0.441949     -0.283232
-0.441949     -0.283232               -0.285378     -0.310382
-0.441949     -0.283232               -0.441933     -0.284444
-0.285378     -0.310382               -0.281697     -0.309632
-0.285378     -0.310382               -0.038464     -0.353198
-0.281697     -0.309632               -0.323825     -0.302345
-0.281697     -0.309632               -0.229838     -0.299069
-0.281697     -0.309632               -0.281927     -0.281998
-0.038464     -0.353198                0.105424     -0.358930
-0.038464     -0.353198               -0.038462     -0.354563
-0.323825     -0.302345               -0.324085     -0.275055
-0.331328     -0.273847               -0.331747     -0.230833
-0.331328     -0.273847               -0.281982     -0.281971
-0.331747     -0.230833               -0.272387     -0.220590
-0.331747     -0.230833               -0.282344     -0.238315
-0.272387     -0.220590               -0.223144     -0.227762
-0.222864     -0.270512               -0.223144     -0.227762
-0.222864     -0.270512               -0.281982     -0.281971
-0.229838     -0.299069               -0.230021     -0.271942
-0.223144     -0.227762               -0.282344     -0.238315
-0.282344     -0.238315               -0.281982     -0.281971
 0.486590     -0.481958                0.003725     -0.419425
 0.486590     -0.481958                0.520359     -0.405502
 0.003725     -0.419425               -0.558043     -0.310944
 0.003725     -0.419425                0.003731     -0.356262
-0.558043     -0.310944               -0.563476     -0.247396
 0.484870     -0.417315                0.105420     -0.360307
 0.105420     -0.360307               -0.038462     -0.354563
-0.038462     -0.354563               -0.441933     -0.284444
-0.423097     -0.279767               -0.497688     -0.265658
-0.469584     -0.259588               -0.565043     -0.247193
 0.195087     -0.310083                0.157483     -0.323920
 0.105424     -0.358930                0.105420     -0.360307
```



CARRIER (1)

FIGURE 35

```
   PARAMETERS FOR THIS VIEW

OBJECT POSITION (1)                        126.0
OBJECT POSITION (2)                          0.0
OBJECT POSITION (3)                          0.0
OBJECT ANGLE ALPHA                           0.0
OBJECT ANGLE BETA                            0.0
OBJECT ANGLE GAMMA                           0.0
OBSERVERS POSITION (1)                    1914.0
OBSERVERS POSITION (2)                     151.0
OBSERVERS POSITION (3)                     112.5
OBSERVERS ANGLE PSI                         10.0
OBSERVERS ANGLE THETA                        0.0
OBSERVERS ANGLE PHI                          0.0
FOCAL LENGTH                                 1.5
SCALE                                        0.5
HORIZONTAL OFFSET                            0.0
VERTICAL OFFSET                              0.0
```

TABLE V

PARAMETERS AND STRUCTURED DATA FOR CARRIER (2)

```
COLUMN      1          2           3          4             5           6       7        13
IDENT    THREE-DIMENSICNAL COORDINATES      DEPTH       PICTURE PLANE COORD CCV/CVX NATURE
  1         0.0        45.6000     57.0000 -1779.1384     0.3482     -0.0935     0        1  1
  2         0.0         0.0        57.0000 -1787.0569     0.2714     -0.0931     0        1  2
  3         0.0       -45.6000     57.0000 -1794.9751     0.1952     -0.0927     0        0  5
  4       353.3999    -64.0000     57.0000 -1450.1396     0.0773     -0.1147     1        0  5
  5       365.0000   -131.0000     57.0000 -1450.3501    -0.0633     -0.1147     0        0  5
  6       955.0000   -131.0000     57.0000  -869.3137    -0.4584     -0.1912     0        0  5
  7      1009.0000    -79.8000     57.0000  -807.2432    -0.3414     -0.2059     1        0  5
  8      1068.0000    -68.5000     57.0000  -747.1772    -0.3652     -0.2224     0        0  5
  9      1046.0000     62.7000     57.0000  -746.0603     0.1681     -0.2227     0        0  5
 10      1052.0000     45.6000     57.0000  -743.1211     0.0967     -0.2236     0        0  5
 11       353.0000    -64.0000     57.0000 -1450.5334     0.0774     -0.1147     0        0  5
 12      1058.5000      5.7000     57.0000  -743.6482    -0.0661     -0.2234     0        0  5
 13       358.5000   -103.9000     57.0000 -1452.0457    -0.0058     -0.1145     0        0  5
 14      1065.0000    -33.5000     57.0000  -744.0542    -0.2259     -0.2233     0        0  5
 15       450.5000   -131.0000     57.0000 -1366.1492    -0.0997     -0.1217     0        0  5
 16       930.0000     50.2000     57.0000  -862.4688     0.1727     -0.1927     1        0  5
 17       930.0000     85.6000     57.0000  -856.3215     0.2958     -0.1941     0        0  5
 18       828.0000     90.0000     57.0000  -956.0081     0.3341     -0.1739     1        0  5
 19       828.0000    114.0000     57.0000  -951.8403     0.4099     -0.1746     0        0  5
 20       647.0000    114.0000     57.0000 -1130.0906     0.4287     -0.1471     0        0  5
 21       647.0000    110.0000     57.0000 -1130.7852     0.4180     -0.1470     1        0  5
 22       694.0000    110.0000     57.0000 -1084.4990     0.4132     -0.1533     1        0  5
 23       694.0000     52.5000     57.0000 -1094.4839     0.2545     -0.1519     1        0  6
 24       647.0000     52.5000     57.0000 -1140.7700     0.2656     -0.1458     1        0  0
 25       394.0000    114.0000     57.0000 -1379.2468     0.4468     -0.1206     0        0  5
 26       302.0000     63.0000     57.0000 -1478.7053     0.3473     -0.1125     1        1  3
 27       694.0000    110.0000     79.8000 -1084.4990     0.4132     -0.0903     1        1  6
 28       694.0000     52.5000     79.8000 -1094.4839     0.2545     -0.0895     1        1  7
 29       698.0000     48.5000     79.8000 -1091.2393     0.2425     -0.0898     0        0  8
 30       698.0000    114.0000     79.8000 -1079.8652     0.4240     -0.0907     0        0  8
 31       698.0000    114.0000    114.0000 -1079.8652     0.4240      0.0042     0        0  8
 32       698.0000     48.5000    114.0000 -1091.2393     0.2425      0.0041     0        0  8
 33       643.0000     48.5000     79.8000 -1145.4038     0.2560     -0.0855     0        0  0
 34       643.0000     48.5000    114.0000 -1145.4038     0.2560      0.0039     0        0  0
 35       647.0000     52.5000     79.8000 -1140.7700     0.2656     -0.0859     1        0  0
 36       643.0000    114.0000    114.0000 -1134.0298     0.4290      0.0040     0        0  8
 37       643.0000    114.0000     79.8000 -1134.0298     0.4290     -0.0864     0        0  8
 38       647.0000    110.0000     79.8000 -1130.7852     0.4180     -0.0866     1        1  5
 39        25.0000      0.0         0.0    -1762.4365     0.2678     -0.1913     0        0  0
 40       394.0000     66.5000      0.0    -1387.4951     0.3431     -0.2430     0        0  8
```

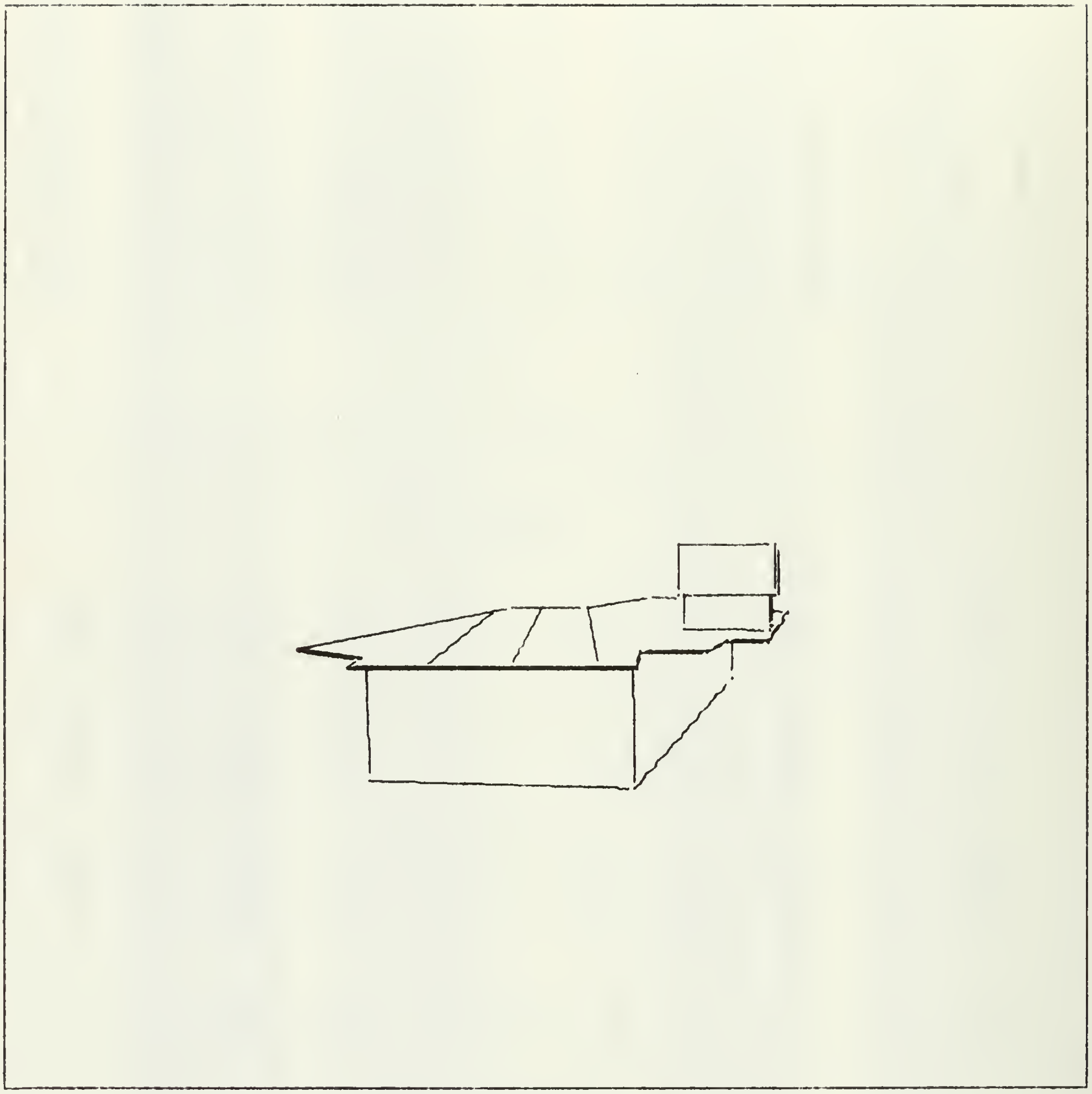
```
COLUMN        1           2              3              4            5             6          7       13
IDENT     THREE-DIMENSIGNAL COORDINATES               DEPTH       PICTURE PLANE COORD  CCV/CVX  NATURE
  41     1040.0000     61.0000         0.0        -752.2644      0.1642       -0.4478     0       0  8
  42     1040.0000    -67.0000         0.0        -774.4915     -0.3278       -0.4349     0       0 10
  43      394.0000    -66.5000         0.0       -1410.5906      0.0592       -0.2390     0       0  0
  44     1046.0000    -67.0000        56.0000     -768.5825     -0.3344       -0.2201     1       1 11
  45     1046.0000     61.0000        56.0000     -746.3555      0.1613       -0.2266     1       1 10
  46      394.0000    -66.5000        56.0000    -1410.5906      0.0592       -0.1200     1       0  0
  47        0.0         0.0           56.0000    -1787.0569      0.2714       -0.0948     1       0  0
  48      394.0000     66.5000        56.0000    -1387.4951      0.3431       -0.1220     1       2  8
  49        0.0        45.6000        56.0000    -1779.1384      0.3482       -0.0952     0       1  4
  50      302.0000     63.0000        56.0000    -1478.7053      0.3473       -0.1145     1       1 12
  51      394.0000    114.0000        56.0000    -1379.2468      0.4468       -0.1228     0       0  7
  52      828.0000    114.0000        56.0000     -951.8403      0.4099       -0.1778     0       0  5
  53      828.0000     90.0000        56.0000     -956.0081      0.3341       -0.1770     1       0  5
  54      930.0000     85.6000        56.0000     -856.3215      0.2958       -0.1976     0       0  5
  55      930.0000     50.2000        56.0000     -862.4688      0.1727       -0.1962     1       0  5
  56     1046.0000     62.7000        56.0000     -746.0603      0.1681       -0.2267     0       0  5
  57     1068.0000    -68.5000        56.0000     -747.1772     -0.3652       -0.2264     0       0  5
  58     1009.0000    -79.8000        56.0000     -807.2432     -0.3413       -0.2096     1       1 13
  59      955.0000   -131.0000        56.0000     -869.3137     -0.4584       -0.1946     0       0  5
  60      365.0000   -131.0000        56.0000    -1450.3501     -0.0633       -0.1167     0       0  0
  61      353.3999    -64.0000        56.0000    -1450.1396      0.0773       -0.1168     1       0  0
  62        0.0       -45.6000        56.0000    -1794.9751      0.1952       -0.0944     0       0  0
  63        0.0         0.0            0.0           0.0         0.0          0.0       0       0  0
  64        0.0         0.0            0.0           0.0         0.0          0.0       0       0  0
  65        0.0         0.0            0.0           0.0         0.0          0.0       0       0  0
  66        0.0         0.0            0.0           0.0         0.0          0.0       0       0  0
  67        0.0         0.0            0.0           0.0         0.0          0.0       0       0  0
  68        0.0         0.0            0.0           0.0         0.0          0.0       0       0  0
  69        0.0         0.0            0.0           0.0         0.0          0.0       0       0  0
  70        0.0         0.0            0.0           0.0         0.0          0.0       0       0  0
  71        0.0         0.0            0.0           0.0         0.0          0.0       0       0  0
  72        0.0         0.0            0.0           0.0         0.0          0.0       0       0  0
  73        0.0         0.0            0.0           0.0         0.0          0.0       0       0  0
  74        0.0         0.0            0.0           0.0         0.0          0.0       0       0  0
  75        0.0         0.0            0.0           0.0         0.0          0.0       0       0  0
  76        0.0         0.0            0.0           0.0         0.0          0.0       0       0  0
  77        0.0         0.0            0.0           0.0         0.0          0.0       0       0  0
  78        0.0         0.0            0.0           0.0         0.0          0.0       0       0  0
  79        0.0         0.0            0.0           0.0         0.0          0.0       0       0  0
  80        0.0         0.0            0.0           0.0         0.0          0.0       0       0  0
```

```
COLUMN    8                 9            10               11            12
IDENT   FACES             LINE END POINTS              INTERSECTING SURFACES
    1    36     1         1    -2         1   -49        1    32       32    19
    2     4    37         2    -3        25   -51        1    32       20    21
    3     4    41         3    -4        19   -52        1    31       21    22
    4    -4    45         4    -5        18   -53        1    30      -22    23
    5    -4    49         5    -6        17   -54        1    29       23    24
    6    -4    53         6    -7        16   -55        1    28      -24    25
    7    -4    57         7    -8         9   -56        1    27       25    26
    8     4    61         8    -9         8   -57        1    26       26    27
    9     4    65         9   -10        -7    58        1     1      -27    28
   10    -4    69        10   -11         6   -59        1     1       28    29
   11    -5    73        10   -12        -5    60        1     1       29    30
   12     4    78        12   -13        -4    61        1     1      -30    31
   13    -4    82        12   -14        -3    62        1     1       31    32
   14    -4    86        14   -15         0     0        1     1        0     0
   15    -4    90         8   -14        -7     0        1     1        1     1
   16     4    94         9   -16       -72     0        1    25       28     2
   17   -20    98        16   -17       -25     0        1    24        2     2
   18   -10   118        17   -18       -68     0        1    23        3     2
   19     4   128        18   -19       -44     0        1    22       16     2
   20    -4   132        19   -20         0     0        1    21        0     0
   21     5   136        20   -21         0     0        1     1        0     0
   22     4   141        21   -22         0     0       -1     9        0     0
   23     4   145        22   -23         0     0       -1     2        0     0
   24     4   149       -23    24         0     0       -1     5        0     0
   25     4   153       -21    24         0     0       -7     1        0     0
   26     4   157        20   -25         0     0        1    21        0     0
   27    -4   161        25   -26         0     0        1    20        0     0
   28     4   165         1   -26         0     0       19     1        0     0
   29    -4   169        22   -27         0     0        2     9        0     0
   30    -4   173       -27    28         0     0       -2    18        0     0
   31    -4   177        23   -28         0     0        5     2        0     0
   32    -6   181        29   -30         0     0        3    18        0     0
   33     0     0        30   -31         0     0        3     8        0     0
   34     0     0        31   -32         0     0        3    10        0     0
   35     0     0        29   -32         0     0        4     3        0     0
   36     0     0       -29    33         0     0       18     4        0     0
   37     0     0       -32    34         0     0        4    10        0     0
   38     0     0       -33    34         0     0        6     4        0     0
   39     0     0       -28    35         0     0       -5    18        0     0
   40     0     0       -24    35         0     0        7     5        0     0
   41     0     0       -34    36         0     0        6    10        0     0
   42     0     0        36   -37         0     0        6     8        0     0
   43     0     0       -33    37         0     0       18     6        0     0
   44     0     0       -35    38         0     0       -7    18        0     0
   45     0     0        21   -38         0     0        9     7        0     0
   46     0     0        30   -37         0     0        8    18        0     0
   47     0     0        31   -36         0     0       10     8        0     0
   48     0     0       -27    38         0     0      -18     9        0     0
   49     0     0       -39    40         0     0       11    15        0     0
   50     0     0        40   -41         0     0       11    16        0     0
   51     0     0        41   -42         0     0       11    12        0     0
   52     0     0       -42    43         0     0       11    13        0     0
   53     0     0       -39    43         0     0       14    11        0     0
   54     0     0        42   -44         0     0       13    12        0     0
   55     0     0        41   -45         0     0       12    16        0     0
   56     0     0       -44    45         0     0      -17    12        0     0
   57     0     0       -43    46         0     0       14    13        0     0
   58     0     0       -44    46         0     0      -13    17        0     0
   59     0     0       -39    47         0     0       15    14        0     0
   60     0     0       -46    47         0     0      -14    17        0     0
   61     0     0       -47    48         0     0      -15    17        0     0
   62     0     0        40   -48         0     0       16    15        0     0
   63     0     0       -45    48         0     0      -17    16        0     0
   64     0     0       -47    49         0     0       17    32        0     0
```

```
COLUMN     8              9          10            11           12
IDENT   FACES           LINE END POINTS        INTERSECTING SURFACES
 65       0     0        49 -50        0      0      17   19        0     0
 66       0     0       -50  51        0      0      17   20        0     0
 67       0     0        51 -52        0      0      17   21        0     0
 68       0     0        52 -53        0      0      17   22        0     0
 69       0     0        53 -54        0      0      17   23        0     0
 70       0     0        54 -55        0      0      17   24        0     0
 71       0     0        55 -56        0      0      17   25        0     0
 72       0     0        56 -57        0      0      17   26        0     0
 73       0     0       -57  58        0      0      17   27        0     0
 74       0     0        58 -59        0      0      17   28        0     0
 75       0     0       -59  60        0      0      17   29        0     0
 76       0     0       -60  61        0      0      17   30        0     0
 77       0     0       -61  62        0      0      17   31        0     0
 78       0     0       -47  62        0      0      32   17        0     0
 79       0     0       -37  38        0      0      18   18        0     0
 80       0     0       -26  50        0      0     -19   20        0     0
```

```
                                  LSTFAC

    1       2       3       4       5              6       7       8       9      10
   11      10      12      13      12             14      15      14       8      14
   12      10       9      16      17             18      19      20      21      22
   23      24      21      20      25             26      23      22      27      28
   29      30      31      32      33             29      32      34      24      23

   28      35      33      34      36             37      24      35      38      21
   30      37      36      31      22             21      38      27      34      32
   31      36      39      40      41             42      43      44      42      41
   45      46      43      42      44             47      39      43      46      39
   47      48      40      40      48             45      41      47      46      44

   45      48      47      49      50             51      52      53      54      55
   56      57      58      59      60             61      62      38      35      28
   27      38      37      30      29             33      37      26      50      49
    1      25      51      50      26             19      52      51      25      20
   18      53      52      19      17             54      53      18      16      55

   54      17       9      56      55             16       8      57      56       9
    7      58      57       8       6             59      58       7       5      60
   59       6       4      61      60              5       3      62      61       4
    3       2       1      49      47             62

                                  LSTBFC

    2       2       2       2       3              3       3       2       1      26
    1       2       1
```

```
                         VISIBLE LINES
    XSTART       YSTART              XEND        YEND
  0.254484    -0.093001          0.195166    -0.092681
  0.195166    -0.092681          0.077256    -0.114698
  0.077256    -0.114698         -0.063259    -0.114681
 -0.063259    -0.114681         -0.458421    -0.191200
 -0.458421    -0.191200         -0.341350    -0.205875
 -0.458421    -0.191200         -0.458421    -0.194646
 -0.341350    -0.205875         -0.365197    -0.222392
 -0.365197    -0.222392          0.168101    -0.222724
 -0.365197    -0.222392         -0.225936    -0.223324
 -0.365197    -0.222392         -0.365197    -0.226399
  0.168101    -0.222724          0.096719    -0.223604
  0.168101    -0.222724          0.172651    -0.192715
  0.168101    -0.222724          0.168101    -0.226737
  0.096719    -0.223604          0.077378    -0.114667
  0.096719    -0.223604         -0.066092    -0.223445
 -0.066092    -0.223445         -0.005773    -0.114548
 -0.066092    -0.223445         -0.225936    -0.223324
 -0.225936    -0.223324         -0.099721    -0.121742
  0.172651    -0.192715          0.295809    -0.194096
  0.172651    -0.192715          0.172651    -0.196187
  0.295809    -0.194096          0.334083    -0.173889
  0.295809    -0.194096          0.295809    -0.197593
  0.334083    -0.173889          0.409921    -0.174649
  0.334083    -0.173889          0.334083    -0.177022
  0.409921    -0.174649          0.428676    -0.147138
  0.409921    -0.174649          0.409921    -0.177796
  0.428676    -0.147138          0.417975    -0.147048
  0.428676    -0.147138          0.446776    -0.120587
  0.417975    -0.147048          0.413244    -0.153315
  0.417975    -0.147048          0.417981    -0.093750
  0.413244    -0.153315          0.254478    -0.151918
  0.413244    -0.153315          0.413250    -0.093750
  0.446776    -0.120587          0.417978    -0.118240
  0.446776    -0.120587          0.446776    -0.122760
  0.254478    -0.151918          0.254484    -0.090088
  0.242512    -0.089774          0.424017    -0.090719
  0.242512    -0.089774          0.242522     0.004118
  0.424017    -0.090719          0.424017     0.004161
  0.424017    -0.090719          0.429023    -0.086391
  0.424017     0.004161          0.242522     0.004118
  0.424017     0.004161          0.429023     0.003963
  0.429023     0.003963          0.429023    -0.086391
  0.343089    -0.242982          0.164201    -0.447752
  0.343089    -0.242982          0.343096    -0.176758
  0.164201    -0.447752         -0.327835    -0.434927
  0.164201    -0.447752          0.161320    -0.226730
 -0.327835    -0.434927         -0.334216    -0.226419
  0.446776    -0.122760          0.409921    -0.177796
  0.409921    -0.177796          0.334083    -0.177022
  0.334083    -0.177022          0.295809    -0.197593
  0.295809    -0.197593          0.172651    -0.196187
  0.172651    -0.196187          0.168101    -0.226737
  0.168101    -0.226737         -0.365197    -0.226399
 -0.345879    -0.209006         -0.458421    -0.194646
```



CARRIER (2)

FIGURE 36

```
   PARAMETERS FOR THIS VIEW

OBJECT POSITION (1)                          84.0
OBJECT POSITION (2)                           0.0
OBJECT POSITION (3)                           0.0
OBJECT ANGLE ALPHA                            0.0
OBJECT ANGLE BETA                             0.0
OBJECT ANGLE GAMMA                            0.0
OBSERVERS POSITION (1)                     1598.0
OBSERVERS POSITION (2)                       96.0
OBSERVERS POSITION (3)                       96.0
OBSERVERS ANGLE PSI                          10.0
OBSERVERS ANGLE THETA                         0.0
OBSERVERS ANGLE PHI                           0.0
FOCAL LENGTH                                  1.5
SCALE                                         0.5
HORIZONTAL OFFSET                             0.0
VERTICAL OFFSET                               0.0
```

TABLE VI

PARAMETERS AND STRUCTURED DATA FOR CARRIER (3)

```
COLUMN         1                 2                 3                4                5                 6          7         13
IDENT      THREE-DIMENSIONAL COORDINATES                         DEPTH          PICTURE PLANE COORD      CCV/CVX   NATURE
   1           0.0          45.6000           57.0000  -1499.7505          0.4262       -0.0779          0        1   1
   2           0.0           0.0              57.0000  -1507.6689          0.3347       -0.0775          0        0   2
   3           0.0         -45.6000           57.0000  -1515.5872          0.2441       -0.0771          0        0   3
   4         353.3999      -64.0000           57.0000  -1170.7517          0.1125       -0.0998          1        0   3
   5         365.0000     -131.0000           57.0000  -1170.9622         -0.0615       -0.0998          0        0   3
   6         955.0000     -131.0000           57.0000   -589.9258         -0.6416       -0.1978          0        0   3
   7        1009.0000      -79.8000           57.0000   -527.8552         -0.4842       -0.2210          1        0   3
   8        1068.0000      -68.5000           57.0000   -467.7893         -0.5405       -0.2493          0        0   3
   9        1046.0000       62.7000           57.0000   -466.6724          0.3106       -0.2499          0        0   3
  10        1052.0000       45.6000           57.0000   -463.7332          0.1973       -0.2515          0        0   3
  11         353.0000      -64.0000           57.0000  -1171.1455          0.1127       -0.0998          0        0   3
  12        1058.5000        5.7000           57.0000   -464.2603         -0.0633       -0.2512          0        0   3
  13         358.5000     -103.9000           57.0000  -1172.6577          0.0097       -0.0996          0        0   3
  14        1065.0000      -33.5000           57.0000   -464.6663         -0.3190       -0.2510          0        0   3
  15         450.5000     -131.0000           57.0000  -1086.7612         -0.1072       -0.1075          0        0   3
  16         930.0000       50.2000           57.0000   -583.0808          0.2890       -0.2001          1        0   3
  17         930.0000       85.6000           57.0000   -576.9336          0.4728       -0.2023          0        0   3
  18         828.0000       90.0000           57.0000   -676.6201          0.5009       -0.1725          1        0   3
  19         828.0000      114.0000           57.0000   -672.4524          0.6092       -0.1736          0        0   3
  20         647.0000      114.0000           57.0000   -850.7026          0.5924       -0.1373          0        0   3
  21         647.0000      110.0000           57.0000   -851.3972          0.5781       -0.1372          1        1   3
  22         694.0000      110.0000           57.0000   -805.1111          0.5809       -0.1451          1        0   5
  23         694.0000       52.5000           57.0000   -815.0959          0.3657       -0.1433          1        0   5
  24         647.0000       52.5000           57.0000   -861.3821          0.3745       -0.1356          1        0   0
  25         394.0000      114.0000           57.0000  -1099.8589          0.5780       -0.1062          0        1   4
  26         302.0000       63.0000           57.0000  -1199.3174          0.4446       -0.0974          1        1   2
  27         694.0000      110.0000           79.8000   -805.1111          0.5809       -0.0603          1        1   5
  28         694.0000       52.5000           79.8000   -815.0959          0.3657       -0.0595          1        1   6
  29         698.0000       48.5000           79.8000   -811.8513          0.3501       -0.0598          0        0   7
  30         698.0000      114.0000           79.8000   -800.4773          0.5964       -0.0606          0        0   7
  31         698.0000      114.0000          114.0000   -800.4773          0.5964        0.0673          0        0   7
  32         698.0000       48.5000          114.0000   -811.8513          0.3501        0.0664          0        0   7
  33         643.0000       48.5000           79.8000   -866.0159          0.3613       -0.0560          0        0   0
  34         643.0000       48.5000          114.0000   -866.0159          0.3613        0.0622          0        0   0
  35         647.0000       52.5000           79.8000   -861.3821          0.3745       -0.0563          1        0   0
  36         643.0000      114.0000          114.0000   -854.6418          0.5921        0.0631          0        0   0
  37         643.0000      114.0000           79.8000   -854.6418          0.5921       -0.0568          0        0   0
  38         647.0000      110.0000           79.8000   -851.3972          0.5781       -0.0570          1        0   0
  39          25.0000        0.0               0.0     -1483.0486          0.3315       -0.1940          0        0   0
  40         394.0000       66.5000            0.0     -1108.1072          0.4473       -0.2596          0        0   7
```

```
COLUMN         1              2              3              4            5           6        7       13
IDENT     THREE-DIMENSIONAL COORDINATES                   DEPTH       PICTURE PLANE COORD  CCV/CVX  NATURE
 41      1040.0000        61.0000         0.0        -472.8765       0.3026     -0.6071      0       0  7
 42      1040.0000       -67.0000         0.0        -495.1035      -0.4725     -0.5799      0       0  9
 43       394.0000       -66.5000         0.0       -1131.2026       0.0913     -0.2543      0       0  0
 44      1046.0000       -67.0000        56.0000     -489.1946      -0.4846     -0.2446      1       1 10
 45      1046.0000        61.0000        56.0000     -466.9675       0.2997     -0.2562      1       1  9
 46       394.0000       -66.5000        56.0000    -1131.2026       0.0913     -0.1059      1       0  0
 47         0.0            0.0           56.0000    -1507.6689       0.3347     -0.0795      1       0  0
 48       394.0000        66.5000        56.0000    -1108.1072       0.4473     -0.1081      1       2  7
 49         0.0           45.6000        56.0000    -1499.7505       0.4262     -0.0799      0       0  0
 50       302.0000        63.0000        56.0000    -1199.3174       0.4446     -0.0999      1       0  0
 51       394.0000       114.0000        56.0000    -1099.8589       0.5780     -0.1090      0       0  0
 52       828.0000       114.0000        56.0000     -672.4524       0.6092     -0.1781      0       0  3
 53       828.0000        90.0000        56.0000     -676.6201       0.5009     -0.1770      1       0  3
 54       930.0000        85.6000        56.0000     -576.9336       0.4728     -0.2075      0       0  3
 55       930.0000        50.2000        56.0000     -583.0808       0.2890     -0.2053      1       1 11
 56      1046.0000        62.7000        56.0000     -466.6724       0.3106     -0.2563      0       0  3
 57      1068.0000       -68.5000        56.0000     -467.7893      -0.5405     -0.2557      0       0  3
 58      1009.0000       -79.8000        56.0000     -527.8552      -0.4842     -0.2267      1       1 12
 59       955.0000      -131.0000        56.0000     -589.9258      -0.6416     -0.2029      0       0  3
 60       365.0000      -131.0000        56.0000    -1170.9622      -0.0615     -0.1023      0       0  0
 61       353.3999       -64.0000        56.0000    -1170.7517       0.1125     -0.1024      1       0  0
 62         0.0          -45.6000        56.0000    -1515.5872       0.2441     -0.0791      0       0  0
 63         0.0            0.0            0.0           0.0          0.0         0.0         0       0  0
 64         0.0            0.0            0.0           0.0          0.0         0.0         0       0  0
 65         0.0            0.0            0.0           0.0          0.0         0.0         0       0  0
 66         0.0            0.0            0.0           0.0          0.0         0.0         0       0  0
 67         0.0            0.0            0.0           0.0          0.0         0.0         0       0  0
 68         0.0            0.0            0.0           0.0          0.0         0.0         0       0  0
 69         0.0            0.0            0.0           0.0          0.0         0.0         0       0  0
 70         0.0            0.0            0.0           0.0          0.0         0.0         0       0  0
 71         0.0            0.0            0.0           0.0          0.0         0.0         0       0  0
 72         0.0            0.0            0.0           0.0          0.0         0.0         0       0  0
 73         0.0            0.0            0.0           0.0          0.0         0.0         0       0  0
 74         0.0            0.0            0.0           0.0          0.0         0.0         0       0  0
 75         0.0            0.0            0.0           0.0          0.0         0.0         0       0  0
 76         0.0            0.0            0.0           0.0          0.0         0.0         0       0  0
 77         0.0            0.0            0.0           0.0          0.0         0.0         0       0  0
 78         0.0            0.0            0.0           0.0          0.0         0.0         0       0  0
 79         0.0            0.0            0.0           0.0          0.0         0.0         0       0  0
 80         0.0            0.0            0.0           0.0          0.0         0.0         0       0  0
```

```
COLUMN     8                9           10                11            12
IDENT    FACES          LINE END POINTS           INTERSECTING SURFACES
     1     36    1        1   -2      -1   49         1   32       32   19
     2      4   37        2   -3     -25   51         1   32       20   21
     3      4   41        3   -4      19  -52         1   31       21   22
     4     -4   45        4   -5      18  -53         1   30      -22   23
     5     -4   49        5   -6      17  -54         1   29       23   24
     6     -4   53        6   -7     -16   55         1   28      -24   25
     7     -4   57        7   -8       9  -56         1   27       25   26
     8     -4   61        8   -9       8  -57         1   26       26   27
     9     -4   65        9  -10      -7   58         1    1      -27   28
    10     -4   69       10  -11       6  -59         1    1       28   29
    11     -5   73       10  -12      -5   60         1    1       29   30
    12      4   78       12  -13      -4   61         1    1      -30   31
    13     -4   82       12  -14      -3   62         1    1       31   32
    14     -4   86       14  -15       0    0         1    1        0    0
    15     -4   90        8  -14      -7    0         1    1        1    1
    16      4   94        9  -16     -72    0         1   25       28    2
    17    -20   98       16  -17     -25    0         1   24        2    2
    18    -10  118       17  -18     -70    0         1   23        3    2
    19     -4  128       18  -19     -44    0         1   22       16    2
    20     -4  132       19  -20       0    0         1   21        0    0
    21     -5  136       20  -21       0    0         1    1        0    0
    22      4  141      -21   22       0    0        -1    9        0    0
    23      4  145       22  -23       0    0        -1    2        0    0
    24      4  149      -23   24       0    0        -1    5        0    0
    25     -4  153      -21   24       0    0        -7    1        0    0
    26      4  157       20  -25       0    0         1   21        0    0
    27     -4  161       25  -26       0    0         1   20        0    0
    28      4  165        1  -26       0    0        19    1        0    0
    29     -4  169       22  -27       0    0         2    9        0    0
    30     -4  173      -27   28       0    0        -2   18        0    0
    31     -4  177       23  -28       0    0         5    2        0    0
    32     -6  181       29  -30       0    0         3   18        0    0
    33      0    0       30  -31       0    0         3    8        0    0
    34      0    0       31  -32       0    0         3   10        0    0
    35      0    0       29  -32       0    0         4    3        0    0
    36      0    0      -29   33       0    0        18    4        0    0
    37      0    0      -32   34       0    0         4   10        0    0
    38      0    0      -33   34       0    0         6    4        0    0
    39      0    0      -28   35       0    0        -5   18        0    0
    40      0    0      -24   35       0    0         7    5        0    0
    41      0    0      -34   36       0    0         6   10        0    0
    42      0    0      -36   37       0    0         6    8        0    0
    43      0    0      -33   37       0    0        18    6        0    0
    44      0    0      -35   38       0    0        -7   18        0    0
    45      0    0      -21   38       0    0         9    7        0    0
    46      0    0      -30   37       0    0         8   18        0    0
    47      0    0      -31   36       0    0        10    8        0    0
    48      0    0      -27   38       0    0       -18    9        0    0
    49      0    0      -39   40       0    0        11   15        0    0
    50      0    0       40  -41       0    0        11   16        0    0
    51      0    0       41  -42       0    0        11   12        0    0
    52      0    0      -42   43       0    0        11   13        0    0
    53      0    0      -39   43       0    0        14   11        0    0
    54      0    0       42  -44       0    0        13   12        0    0
    55      0    0       41  -45       0    0        12   16        0    0
    56      0    0      -44   45       0    0       -17   12        0    0
    57      0    0      -43   46       0    0        14   13        0    0
    58      0    0      -44   46       0    0       -13   17        0    0
    59      0    0      -39   47       0    0        15   14        0    0
    60      0    0      -46   47       0    0       -14   17        0    0
    61      0    0      -47   48       0    0       -15   17        0    0
    62      0    0       40  -48       0    0        16   15        0    0
    63      0    0      -45   48       0    0       -17   16        0    0
    64      0    0      -47   49       0    0        17   32        0    0
```

```
COLUMN      8          9         10          11          12
IDENT   FACES       LINE END POINTS      INTERSECTING SURFACES
 65       0    0    -49  50     0    0     17  19       0    0
 66       0    0    -50  51     0    0     17  20       0    0
 67       0    0    -51  52     0    0     17  21       0    0
 68       0    0     52 -53     0    0     17  22       0    0
 69       0    0     53 -54     0    0     17  23       0    0
 70       0    0     54 -55     0    0     17  24       0    0
 71       0    0    -55  56     0    0     17  25       0    0
 72       0    0     56 -57     0    0     17  26       0    0
 73       0    0    -57  58     0    0     17  27       0    0
 74       0    0     58 -59     0    0     17  28       0    0
 75       0    0    -59  60     0    0     17  29       0    0
 76       0    0    -60  61     0    0     17  30       0    0
 77       0    0    -61  62     0    0     17  31       0    0
 78       0    0    -47  62     0    0     32  17       0    0
 79       0    0    -37  38     0    0     18  18       0    0
 80       0    0    -26  50     0    0    -19  20       0    0
```

```
                               LSTFAC

   1     2     3     4     5           6     7     8     9    10
  11    10    12    13    12          14    15    14     8    14
  12    10     9    16    17          18    19    20    21    22
  23    24    21    20    25          26    23    22    27    28
  29    30    31    32    33          29    32    34    24    23

  28    35    33    34    36          37    24    35    38    21
  30    37    36    31    22          21    38    27    34    32
  31    36    39    40    41          42    43    44    42    41
  45    46    43    42    44          47    39    43    46    39
  47    48    40    40    48          45    41    47    46    44

  45    48    47    49    50          51    52    53    54    55
  56    57    58    59    60          61    62    38    35    28
  27    38    37    30    29          33    37    26    50    49
   1    25    51    50    26          19    52    51    25    20
  18    53    52    19    17          54    53    18    16    55

  54    17     9    56    55          16     8    57    56     9
   7    58    57     8     6          59    58     7     5    60
  59     6     4    61    60           5     3    62    61     4
   3     2     1    49    47          62

                               LSTBFC

   2     2     2     2     3           3     2     1    26     1
   1     1
```

```
                          VISIBLE LINES

   XSTART       YSTART                  XEND         YEND
  0.365741    -0.077665               0.334678    -0.077526
  0.334678    -0.077526               0.244129    -0.077121
  0.244129    -0.077121               0.112519    -0.099808
  0.112519    -0.099808              -0.061485    -0.099790
 -0.061485    -0.099790              -0.641578    -0.197827
 -0.641578    -0.197827              -0.484194    -0.221024
 -0.641578    -0.197827              -0.641578    -0.202900
 -0.484194    -0.221024              -0.540522    -0.249313
 -0.540522    -0.249313               0.310611    -0.249908
 -0.540522    -0.249313              -0.318972    -0.250984
 -0.540522    -0.249313              -0.540522    -0.255706
  0.310611    -0.249908               0.197263    -0.251487
  0.310611    -0.249908               0.288957    -0.200143
  0.310611    -0.249908               0.310611    -0.256316
  0.197263    -0.251487               0.112659    -0.099774
  0.197263    -0.251487              -0.063325    -0.251202
 -0.063325    -0.251202               0.009677    -0.099646
 -0.063325    -0.251202              -0.318972    -0.250984
 -0.318972    -0.250984              -0.107170    -0.107511
  0.288957    -0.200143               0.472838    -0.202270
  0.472838    -0.202270               0.500857    -0.172536
  0.472838    -0.202270               0.472838    -0.207457
  0.500857    -0.172536               0.609164    -0.173603
  0.500857    -0.172536               0.500857    -0.176960
  0.609164    -0.173603               0.592394    -0.137291
  0.609164    -0.173603               0.609164    -0.178054
  0.592394    -0.137291               0.580872    -0.137201
  0.592394    -0.137291               0.580874    -0.112350
  0.580871    -0.145051               0.365735    -0.143278
  0.580871    -0.145051               0.580879    -0.066406
  0.365735    -0.143278               0.365743    -0.059082
  0.350102    -0.059753               0.596365    -0.060600
  0.350102    -0.059753               0.350112     0.066392
  0.596365    -0.060600               0.596365     0.067334
  0.596365     0.067334               0.350112     0.066392
  0.447278    -0.259552               0.302550    -0.607113
  0.447278    -0.259552               0.447283    -0.214844
  0.302550    -0.607113              -0.472497    -0.579940
  0.302550    -0.607113               0.299695    -0.256287
 -0.472497    -0.579940              -0.484153    -0.255748
  0.609164    -0.178054               0.500857    -0.176960
  0.500857    -0.176960               0.472838    -0.207457
  0.472838    -0.207457               0.291202    -0.205302
  0.310611    -0.256316              -0.540522    -0.255706
 -0.492874    -0.225379              -0.641578    -0.202900
```

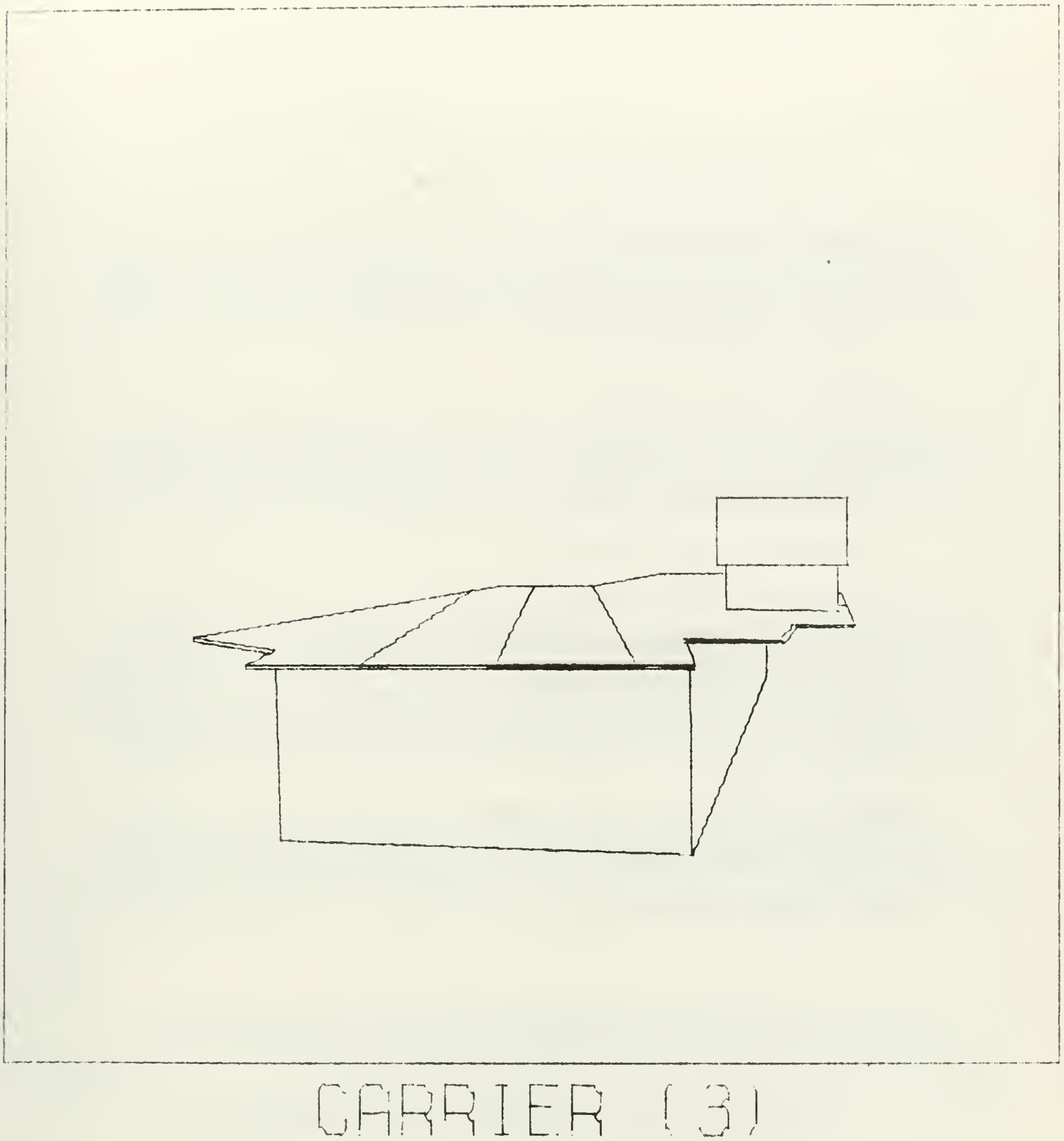


FIGURE 37

```
   PARAMETERS FOR THIS VIEW

OBJECT POSITION (1)                         42.0
OBJECT POSITION (2)                          0.0
OBJECT POSITION (3)                          0.0
OBJECT ANGLE ALPHA                           0.0
OBJECT ANGLE BETA                            0.0
OBJECT ANGLE GAMMA                           0.0
OBSERVERS POSITION (1)                    1300.0
OBSERVERS POSITION (2)                      43.5
OBSERVERS POSITION (3)                      80.0
OBSERVERS ANGLE PSI                         10.0
OBSERVERS ANGLE THETA                        0.0
OBSERVERS ANGLE PHI                          0.0
FOCAL LENGTH                                 1.5
SCALE                                        0.5
HORIZONTAL OFFSET                            0.0
VERTICAL OFFSET                              0.0
```

TABLE VII

PARAMETERS AND STRUCTURED DATA FOR CARRIER (4)

```
COLUMN        1           2           3            4              5            6       7        13
IDENT    THREE-DIMENSIONAL COORDINATES            DEPTH        PICTURE PLANE COORD  CCV/CVX  NATURE
   1        0.0       45.6000     57.0000  -1238.5232        0.5335      -0.0556      0       0  1
   2        0.0        0.0        57.0000  -1246.4417        0.4222      -0.0553      0       0  1
   3        0.0      -45.6000     57.0000  -1254.3599        0.3122      -0.0549      0       0  2
   4      353.3999   -64.0000     57.0000   -909.5244        0.1687      -0.0757      1       0  2
   5      365.0000  -131.0000     57.0000   -909.7349       -0.0552      -0.0757      0       0  2
   6      955.0000  -131.0000     57.0000   -328.6985       -1.0833      -0.2090      0       0  2
   7     1009.0000   -79.8000     57.0000   -266.6279       -0.8748      -0.2573      1       0  2
   8     1068.0000   -68.5000     57.0000   -206.5620       -1.1146      -0.3316      0       0  2
   9     1046.0000    62.7000     57.0000   -205.4451        0.8078      -0.3334      0       0  2
  10     1052.0000    45.6000     57.0000   -202.5059        0.5564      -0.3382      0       0  2
  11      353.0000   -64.0000     57.0000   -909.9182        0.1688      -0.0757      0       0  2
  12     1058.5000     5.7000     57.0000   -203.0330       -0.0379      -0.3374      0       0  2
  13      358.5000  -103.9000     57.0000   -911.4304        0.0363      -0.0756      0       0  2
  14     1065.0000   -33.5000     57.0000   -203.4390       -0.6194      -0.3367      0       0  2
  15      450.5000  -131.0000     57.0000   -825.5339       -0.1147      -0.0834      0       0  2
  16      930.0000    50.2000     57.0000   -321.8535        0.5896      -0.2134      1       0  2
  17      930.0000    85.6000     57.0000   -315.7063        0.9308      -0.2175      0       0  2
  18      828.0000    90.0000     57.0000   -415.3928        0.8669      -0.1655      1       0  2
  19      828.0000   114.0000     57.0000   -411.2251        1.0474      -0.1672      0       0  2
  20      647.0000   114.0000     57.0000   -589.4753        0.8910      -0.1168      0       1  2
  21      647.0000   110.0000     57.0000   -590.1699        0.8700      -0.1166      1       1  3
  22      694.0000   110.0000     57.0000   -543.8838        0.8990      -0.1265      1       0  5
  23      694.0000    52.5000     57.0000   -553.8687        0.5769      -0.1242      1       0  5
  24      647.0000    52.5000     57.0000   -600.1548        0.5732      -0.1147      1       0  5
  25      394.0000   114.0000     57.0000   -838.6316        0.7837      -0.0821      0       1  4
  26      302.0000    63.0000     57.0000   -938.0901        0.5914      -0.0734      1       1  1
  27      694.0000   110.0000     79.8000   -543.8838        0.8990      -0.0011      1       0  5
  28      694.0000    52.5000     79.8000   -553.8687        0.5769      -0.0011      1       0  5
  29      698.0000    48.5000     79.8000   -550.6240        0.5551      -0.0011      0       0  5
  30      698.0000   114.0000     79.8000   -539.2500        0.9247      -0.0011      0       0  5
  31      698.0000   114.0000    114.0000   -539.2500        0.9247       0.1886      0       0  5
  32      698.0000    48.5000    114.0000   -550.6240        0.5551       0.1847      0       0  5
  33      643.0000    48.5000     79.8000   -604.7886        0.5528      -0.0010      0       0  5
  34      643.0000    48.5000    114.0000   -604.7886        0.5528       0.1682      0       0  5
  35      647.0000    52.5000     79.8000   -600.1548        0.5732      -0.0010      1       1  5
  36      643.0000   114.0000    114.0000   -593.4146        0.8886       0.1715      0       0  0
  37      643.0000   114.0000     79.8000   -593.4146        0.8886      -0.0010      0       0  0
  38      647.0000   110.0000     79.8000   -590.1699        0.8700      -0.0010      1       0  0
  39       25.0000     0.0         0.0    -1221.8213        0.4200      -0.1962      0       0  0
  40      394.0000    66.5000      0.0     -846.8799        0.6106      -0.2829      0       0  0
```

```
COLUMN       1               2             3              4              5              6         7        13
IDENT     THREE-DIMENSIONAL COORDINATES                DEPTH       PICTURE PLANE COORD    CCV/CVX  NATURE
  41     1040.0000        61.0000       0.0         -211.6492       0.7754        -1.1260        0        0  6
  42     1040.0000       -67.0000       0.0         -233.8762      -0.9045        -1.0196        0        0  6
  43      394.0000       -66.5000       0.0         -869.9753       0.1436        -0.2754        0        0  0
  44     1046.0000       -67.0000      56.0000      -227.9673      -0.9414        -0.3138        1        1  7
  45     1046.0000        61.0000      56.0000      -205.7402       0.7824        -0.3474        1        1  6
  46      394.0000       -66.5000      56.0000      -869.9753       0.1436        -0.0826        1        0  0
  47        0.0            0.0         56.0000     -1246.4417       0.4222        -0.0577        1        0  0
  48      394.0000        66.5000      56.0000      -846.8799       0.6106        -0.0849        1        0  0
  49        0.0           45.6000      56.0000     -1238.5232       0.5335        -0.0581        0        0  0
  50      302.0000        63.0000      56.0000      -938.0901       0.5914        -0.0766        1        0  0
  51      394.0000       114.0000      56.0000      -838.6316       0.7837        -0.0857        0        0  0
  52      828.0000       114.0000      56.0000      -411.2251       1.0474        -0.1745        0        0  3
  53      828.0000        90.0000      56.0000      -415.3928       0.8669        -0.1727        1        1  8
  54      930.0000        85.6000      56.0000      -315.7063       0.9308        -0.2270        0        0  2
  55      930.0000        50.2000      56.0000      -321.8535       0.5896        -0.2227        1        1  9
  56     1046.0000        62.7000      56.0000      -205.4451       0.8078        -0.3479        0        0  2
  57     1068.0000       -68.5000      56.0000      -206.5620      -1.1146        -0.3460        0        0  2
  58     1009.0000       -79.8000      56.0000      -266.6279      -0.8748        -0.2685        1        1 10
  59      955.0000      -131.0000      56.0000      -328.6985      -1.0833        -0.2181        0        0  2
  60      365.0000      -131.0000      56.0000      -909.7349      -0.0552        -0.0790        0        0  0
  61      353.3999       -64.0000      56.0000      -909.5244       0.1687        -0.0790        1        0  0
  62        0.0          -45.6000      56.0000     -1254.3599       0.3122        -0.0573        0        0  0
  63        0.0            0.0          0.0            0.0          0.0            0.0          0        0  0
  64        0.0            0.0          0.0            0.0          0.0            0.0          0        0  0
  65        0.0            0.0          0.0            0.0          0.0            0.0          0        0  0
  66        0.0            0.0          0.0            0.0          0.0            0.0          0        0  0
  67        0.0            0.0          0.0            0.0          0.0            0.0          0        0  0
  68        0.0            0.0          0.0            0.0          0.0            0.0          0        0  0
  69        0.0            0.0          0.0            0.0          0.0            0.0          0        0  0
  70        0.0            0.0          0.0            0.0          0.0            0.0          0        0  0
  71        0.0            0.0          0.0            0.0          0.0            0.0          0        0  0
  72        0.0            0.0          0.0            0.0          0.0            0.0          0        0  0
  73        0.0            0.0          0.0            0.0          0.0            0.0          0        0  0
  74        0.0            0.0          0.0            0.0          0.0            0.0          0        0  0
  75        0.0            0.0          0.0            0.0          0.0            0.0          0        0  0
  76        0.0            0.0          0.0            0.0          0.0            0.0          0        0  0
  77        0.0            0.0          0.0            0.0          0.0            0.0          0        0  0
  78        0.0            0.0          0.0            0.0          0.0            0.0          0        0  0
  79        0.0            0.0          0.0            0.0          0.0            0.0          0        0  0
  80        0.0            0.0          0.0            0.0          0.0            0.0          0        0  0
```

```
COLUMN        8                   9             10              11              12
IDENT   FACES            LINE END POINTS               INTERSECTING SURFACES
    1     36    1         1   -2        -1   49          1   32         32   19
    2      4   37         2   -3       -25   51          1   32         20   21
    3      4   41         3   -4        19  -52          1   31         21   22
    4      4   45         4   -5       -18   53          1   30        -22   23
    5      4   49         5   -6        17  -54          1   29         23   24
    6     -4   53         6   -7       -16   55          1   28        -24   25
    7     -4   57         7   -8         9  -56          1   27         25   26
    8     -4   61         8   -9         8  -57          1   26         26   27
    9     -4   65         9  -10        -7   58          1    1        -27   28
   10     -4   69        10  -11         6  -59          1    1         28   29
   11     -5   73        10  -12        -5   60          1    1         29   30
   12      4   78        12  -13        -4   61          1    1        -30   31
   13     -4   82        12  -14        -3   62          1    1         31   32
   14     -4   86        14  -15         0    0          1    1          0    0
   15     -4   90         8  -14        -7    0          1    1          1    1
   16     -4   94         9  -16       -72    0          1   25         28    2
   17    -20   98        16  -17       -25    0          1   24         24    2
   18    -10  118        17  -18       -70    0          1   23          2    2
   19     -4  128        18  -19       -44    0          1   22          3    2
   20     -4  132        19  -20         0    0          1   21         12    2
   21     -5  136        20  -21         0    0          1    1          0    0
   22      4  141       -21   22         0    0         -1    9          0    0
   23     -4  145        22  -23         0    0         -1    2          0    0
   24      4  149        23  -24         0    0         -1    5          0    0
   25     -4  153       -21   24         0    0         -7    1          0    0
   26      4  157        20  -25         0    0          1   21          0    0
   27     -4  161        25  -26         0    0          1   20          0    0
   28      4  165         1  -26         0    0         19    1          0    0
   29     -4  169        22  -27         0    0          2    9          0    0
   30     -4  173       -27   28         0    0         -2   18          0    0
   31     -4  177        23  -28         0    0          5    2          0    0
   32     -6  181        29  -30         0    0          3   18          0    0
   33      0    0        30  -31         0    0          3    8          0    0
   34      0    0        31  -32         0    0          3   10          0    0
   35      0    0        29  -32         0    0          4    3          0    0
   36      0    0        29  -33         0    0         18    4          0    0
   37      0    0        32  -34         0    0          4   10          0    0
   38      0    0        33  -34         0    0          6    4          0    0
   39      0    0       -28   35         0    0         -5   18          0    0
   40      0    0        24  -35         0    0          7    5          0    0
   41      0    0       -34   36         0    0          6   10          0    0
   42      0    0       -36   37         0    0          6    8          0    0
   43      0    0       -33   37         0    0         18    6          0    0
   44      0    0       -35   38         0    0         -7   18          0    0
   45      0    0       -21   38         0    0          9    7          0    0
   46      0    0       -30   37         0    0          8   18          0    0
   47      0    0       -31   36         0    0         10    8          0    0
   48      0    0       -27   38         0    0        -18    9          0    0
   49      0    0       -39   40         0    0         11   15          0    0
   50      0    0       -40   41         0    0         11   16          0    0
   51      0    0        41  -42         0    0         11   12          0    0
   52      0    0       -42   43         0    0         11   13          0    0
   53      0    0       -39   43         0    0         14   11          0    0
   54      0    0        42  -44         0    0         13   12          0    0
   55      0    0        41  -45         0    0         12   16          0    0
   56      0    0       -44   45         0    0        -17   12          0    0
   57      0    0       -43   46         0    0         14   13          0    0
   58      0    0       -44   46         0    0        -13   17          0    0
   59      0    0       -39   47         0    0         15   14          0    0
   60      0    0       -46   47         0    0        -14   17          0    0
   61      0    0       -47   48         0    0        -15   17          0    0
   62      0    0       -40   48         0    0         16   15          0    0
   63      0    0       -45   48         0    0        -17   16          0    0
   64      0    0       -47   49         0    0         17   32          0    0
```

```
COLUMN    8          9          10          11          12
IDENT  FACES      LINE END POINTS     INTERSECTING SURFACES
 65     0   0    -49  50       0   0     17  19      0   0
 66     0   0    -50  51       0   0     17  20      0   0
 67     0   0    -51  52       0   0     17  21      0   0
 68     0   0     52 -53       0   0     17  22      0   0
 69     0   0    -53  54       0   0     17  23      0   0
 70     0   0     54 -55       0   0     17  24      0   0
 71     0   0    -55  56       0   0     17  25      0   0
 72     0   0     56 -57       0   0     17  26      0   0
 73     0   0    -57  58       0   0     17  27      0   0
 74     0   0     58 -59       0   0     17  28      0   0
 75     0   0    -59  60       0   0     17  29      0   0
 76     0   0    -60  61       0   0     17  30      0   0
 77     0   0    -61  62       0   0     17  31      0   0
 78     0   0    -47  62       0   0     32  17      0   0
 79     0   0    -37  38       0   0     18  18      0   0
 80     0   0    -26  50       0   0    -19  20      0   0
```

```
                          LSTFAC

    1    2    3    4    5         6    7    8    9   10
   11   10   12   13   12        14   15   14    8   14
   12   10    9   16   17        18   19   20   21   22
   23   24   21   20   25        26   23   22   27   28
   29   30   31   32   33        29   32   34   24   23

   28   35   33   34   36        37   24   35   38   21
   30   37   36   31   22        21   38   27   34   32
   31   36   39   40   41        42   43   44   42   41
   45   46   43   42   44        47   39   43   46   39
   47   48   40   40   48        45   41   47   46   44

   45   48   47   49   50        51   52   53   54   55
   56   57   58   59   60        61   62   38   35   28
   27   38   37   30   29        33   37   26   50   49
    1   25   51   50   26        19   52   51   25   20
   18   53   52   19   17        54   53   18   16   55

   54   17    9   56   55        16    8   57   56    9
    7   58   57    8    6        59   58    7    5   60
   59    6    4   61   60         5    3   62   61    4
    3    2    1   49   47        62

                          LSTBFC

    2    2    2    2    3        26    1    1    1    1
```

VISIBLE LINES

```
      XSTART        YSTART                  XEND        YEND
    0.533501     -0.055644              0.422161   -0.055291
    0.533501     -0.055644              0.573236   -0.067863
    0.422161     -0.055291              0.312225   -0.054942
    0.312225     -0.054942              0.168653   -0.075739
    0.168653     -0.075739             -0.055247   -0.075722
   -0.055247     -0.075722             -1.000000   -0.198170
   -1.000000     -0.228292             -0.874824   -0.257340
   -0.874824     -0.257340             -1.000000   -0.296117
   -1.000000     -0.331737              0.807774   -0.333421
   -1.000000     -0.332801             -0.619443   -0.336686
    0.807774     -0.333421              0.556450   -0.338226
    0.807774     -0.333421              0.589647   -0.213388
    0.807774     -0.333421              0.807774   -0.347917
    0.556450     -0.338226              0.168809   -0.075706
    0.556450     -0.338226             -0.037884   -0.337354
   -0.037884     -0.337354              0.036266   -0.075581
   -0.037884     -0.337354             -0.619443   -0.336686
   -0.619443     -0.336686             -0.114727   -0.083431
    0.589647     -0.213388              0.930785   -0.217524
    0.930785     -0.217524              0.866858   -0.165510
    0.930785     -0.217524              0.930785   -0.226981
    0.866858     -0.165510              1.000000   -0.166743
    1.000000     -0.151892              0.898968   -0.119312
    0.898968     -0.126516              0.576920   -0.124242
    0.898968     -0.126516              0.898978   -0.001100
    0.576920     -0.124242              0.573232   -0.114684
    0.576920     -0.124242              0.576920   -0.001080
    0.555132     -0.001087              0.924672   -0.001110
    0.555132     -0.001087              0.555142    0.184741
    0.555132     -0.001087              0.552796   -0.000990
    0.924672     -0.001110              0.924672    0.188627
    0.924672      0.188627              0.555142    0.184741
    0.555142      0.184741              0.552796    0.168237
    0.552796     -0.000990              0.552796    0.168237
    0.573232     -0.114684              0.573243   -0.007812
    0.776501     -1.000000              0.782384   -0.347916
   -0.905526     -1.000000             -0.939712   -0.346237
    1.000000     -0.173992              0.875809   -0.172792
    0.930785     -0.226981              0.606904   -0.222884
    0.807774     -0.347917             -1.000000   -0.346161
   -0.895102     -0.263619             -1.000000   -0.238218
```

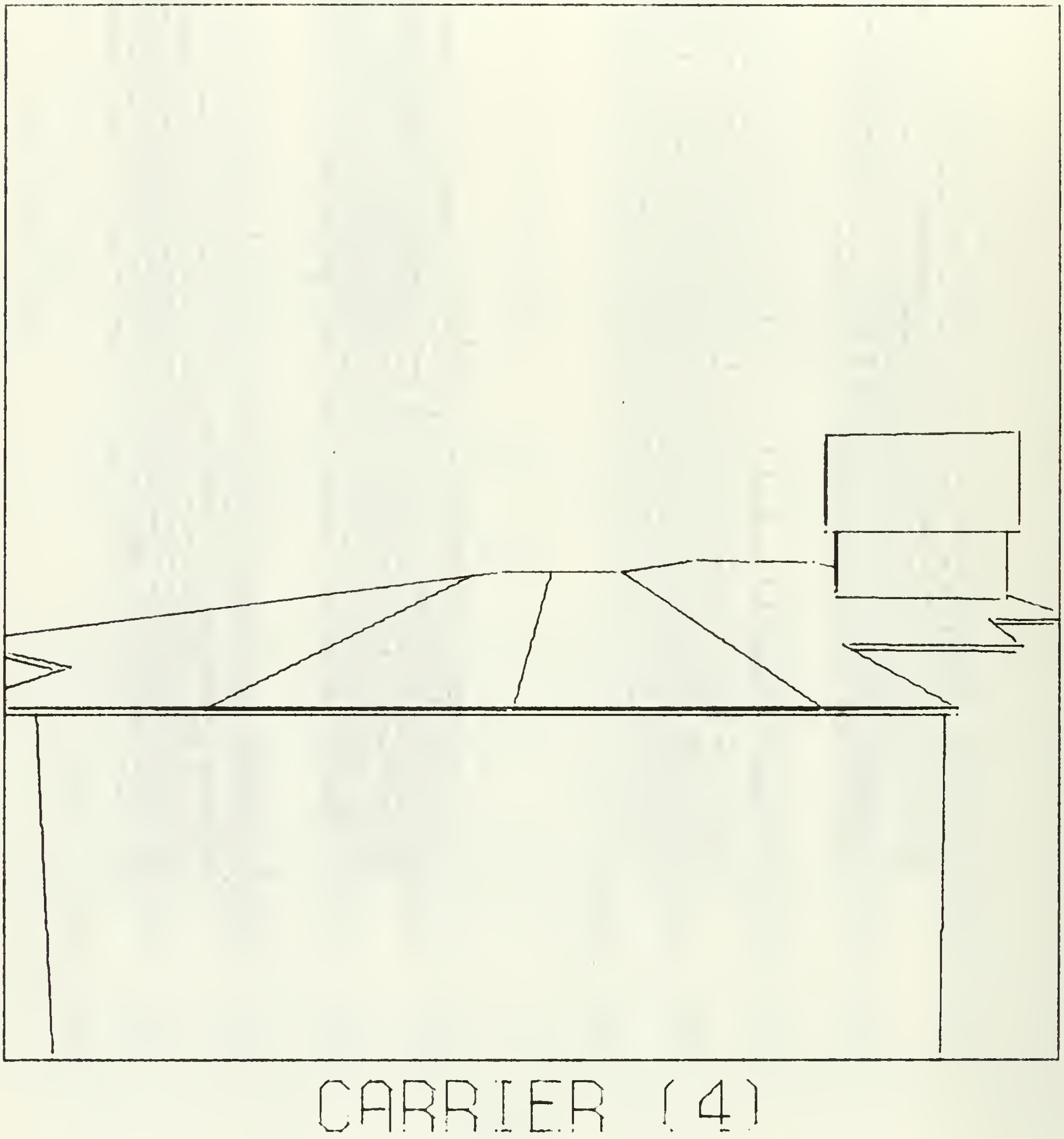


FIGURE 38

```
          PARAMETERS FOR THIS VIEW

      OBJECT POSITION (1)                              0.0
      OBJECT POSITION (2)                              0.0
      OBJECT POSITION (3)                              0.0
      OBJECT ANGLE ALPHA                               0.0
      OBJECT ANGLE BETA                                0.0
      OBJECT ANGLE GAMMA                               0.0
      OBSERVERS POSITION (1)                         950.0
      OBSERVERS POSITION (2)                           0.0
      OBSERVERS POSITION (3)                          66.0
      OBSERVERS ANGLE PSI                             10.0
      OBSERVERS ANGLE THETA                            0.0
      OBSERVERS ANGLE PHI                              0.0
      FOCAL LENGTH                                     1.5
      SCALE                                            0.5
      HORIZONTAL OFFSET                                0.0
      VERTICAL OFFSET                                  0.0
```

TABLE VIII

PARAMETERS AND STRUCTURED DATA FOR CARRIER (5)

```
COLUMN        1             2            3           4             5           6         7      13
IDENT     THREE-DIMENSIONAL COORDINATES            DEPTH       PICTURE PLANE COORD CCV/CVX NATURE
  1          0.0         45.6000      57.0000    -927.6489      0.6776     -0.0291      0      0  1
  2          0.0          0.0         57.0000    -935.5674      0.5281     -0.0288      0      0  1
  3          0.0        -45.6000      57.0000    -943.4856      0.3811     -0.0286      0      0  1
  4        353.3999     -64.0000      57.0000    -598.6501      0.2028     -0.0450      1      0  1
  5        365.0000    -131.0000      57.0000    -598.8606     -0.1370     -0.0450      0      0  1
  6        955.0000    -131.0000      57.0000     -17.8242    -20.1630     -1.3972      0      0  1
  7       1009.0000     -79.8000      57.0000      44.2463      6.2344      0.6316      1      0  0
  8       1068.0000     -68.5000      57.0000     104.3123      2.5663      0.2626      0      0  0
  9       1046.0000      62.7000      57.0000     105.4292     -1.3012      0.2598      0      0  0
 10       1052.0000      45.6000      57.0000     108.3684     -0.7634      0.2526      0      0  0
 11        353.0000     -64.0000      57.0000    -599.0439      0.2030     -0.0450      0      0  1
 12       1058.5000       5.7000      57.0000     107.8413      0.3732      0.2539      0      0  0
 13        358.5000    -103.9000      57.0000    -600.5562      0.0020     -0.0448      0      0  1
 14       1065.0000     -33.5000      57.0000     107.4353      1.4998      0.2549      0      0  0
 15        450.5000    -131.0000      57.0000    -514.6597     -0.2457     -0.0523      0      0  1
 16        930.0000      50.2000      57.0000     -10.9792     12.7191     -2.1635      1      0  1
 17        930.0000      85.6000      57.0000      -4.8320     41.5818     -4.2637      0      0  1
 18        828.0000      90.0000      57.0000    -104.5186      3.1075     -0.2547      1      0  1
 19        828.0000     114.0000      57.0000    -100.3508      3.9309     -0.2651      0      0  1
 20        647.0000     114.0000      57.0000    -278.6011      1.7660     -0.0964      0      1  1
 21        647.0000     110.0000      57.0000    -279.2957      1.7195     -0.0962      1      1  2
 22        694.0000     110.0000      57.0000    -233.0095      1.9545     -0.1151      1      0  4
 23        694.0000      52.5000      57.0000    -242.9944      1.1799     -0.1104      1      0  4
 24        647.0000      52.5000      57.0000    -289.2805      1.0763     -0.0929      1      0  4
 25        394.0000     114.0000      57.0000    -527.7573      1.1836     -0.0510      0      1  3
 26        302.0000      63.0000      57.0000    -627.2158      0.8330     -0.0429      1      0  1
 27        694.0000     110.0000      79.8000    -233.0095      1.9545      0.1765      1      0  4
 28        694.0000      52.5000      79.8000    -242.9944      1.1799      0.1693      1      0  4
 29        698.0000      48.5000      79.8000    -239.7498      1.1381      0.1716      0      0  4
 30        698.0000     114.0000      79.8000    -228.3757      2.0362      0.1801      0      0  4
 31        698.0000     114.0000     114.0000    -228.3757      2.0362      0.6264      0      0  4
 32        698.0000      48.5000     114.0000    -239.7498      1.1381      0.5969      0      0  4
 33        643.0000      48.5000      79.8000    -293.9143      1.0264      0.1401      0      0  4
 34        643.0000      48.5000     114.0000    -293.9143      1.0264      0.4875      0      0  5
 35        647.0000      52.5000      79.8000    -289.2805      1.0763      0.1424      1      0  5
 36        643.0000     114.0000     114.0000    -282.5403      1.7488      0.5070      0      0  0
 37        643.0000     114.0000      79.8000    -282.5403      1.7488      0.1458      0      1  4
 38        647.0000     110.0000      79.8000    -279.2957      1.7195      0.1474      1      1 13
 39         25.0000       0.0          0.0       -910.9470      0.5281     -0.2170      0      0  0
 40        394.0000      66.5000       0.0       -536.0056      0.9044     -0.3684      0      1  5
```

```
COLUMN      1             2            3             4              5            6        7        13
IDENT    THREE-DIMENSIONAL COORDINATES              DEPTH       PICTURE PLANE COORD  CCV/CVX  NATURE
 41      1040.0000      61.0000       0.0          99.2251       -1.3644       2.0261     0      0  0
 42      1040.0000     -67.0000       0.0          76.9980        3.2429       2.6226     0      0  0
 43       394.0000     -66.5000       0.0        -559.1011        0.1662      -0.3532     0      1  9
 44      1046.0000     -67.0000      56.0000       82.9070        3.0459       0.3685     1      0  0
 45      1046.0000      61.0000      56.0000      105.1340       -1.2564       0.2895     1      0  0
 46       394.0000     -66.5000      56.0000     -559.1011        0.1662      -0.0535     1      1 10
 47         0.0          0.0         56.0000     -935.5674        0.5281      -0.0320     1      0  0
 48       394.0000      66.5000      56.0000     -536.0056        0.9044      -0.0558     1      1  7
 49         0.0         45.6000      56.0000     -927.6489        0.6776      -0.0323     0      0  0
 50       302.0000      63.0000      56.0000     -627.2158        0.8330      -0.0477     1      0  0
 51       394.0000     114.0000      56.0000     -527.7573        1.1836      -0.0567     0      0  0
 52       828.0000     114.0000      56.0000     -100.3508        3.9309      -0.2945     0      0  2
 53       828.0000      90.0000      56.0000     -104.5186        3.1075      -0.2830     1      1 11
 54       930.0000      85.6000      56.0000       -4.8320       41.5818      -4.7375     0      0  1
 55       930.0000      50.2000      56.0000      -10.9792       12.7191      -2.4039     1      1 12
 56      1046.0000      62.7000      56.0000      105.4292       -1.3012       0.2887     0      0  0
 57      1068.0000     -68.5000      56.0000      104.3123        2.5663       0.2918     0      0  0
 58      1009.0000     -79.8000      56.0000       44.2463        6.2344       0.7018     1      0  0
 59       955.0000    -131.0000      56.0000      -17.8242      -20.1630      -1.5525     0      0  1
 60       365.0000    -131.0000      56.0000     -598.8606       -0.1370      -0.0500     0      0  0
 61       353.3999     -64.0000      56.0000     -598.6501        0.2028      -0.0500     1      0  0
 62         0.0        -45.6000      56.0000     -943.4856        0.3811      -0.0317     0      0  0
 63       970.5247    -116.2802      57.0000        0.0208     -239.4723     -18.2528     0      0  1
 64       944.9150      28.8096      57.0000       -0.0051       58.3219     -17.9422     0      0  1
 65       951.9390     -10.9844      57.0000        0.0020      -22.3382     -18.0240     0      0  1
 66       958.8850     -50.3368      57.0000        0.0091     -102.8201     -18.1039     0      0  1
 67       940.9314      51.3779      57.0000       -0.0092      103.6908     -17.8873     0      0  1
 68       939.0813      61.8592       0.0          -0.0112      124.7219    -131.0449     0      1  6
 69       961.8154     -66.9395       0.0           0.0118     -137.0495    -133.0690     0      1  8
 70       970.5247    -116.2802      56.0000        0.0208     -239.4723     -20.2809     0      0 13
 71         0.0          0.0          0.0           0.0           0.0          0.0       0      0  0
 72         0.0          0.0          0.0           0.0           0.0          0.0       0      0  0
 73         0.0          0.0          0.0           0.0           0.0          0.0       0      0  0
 74         0.0          0.0          0.0           0.0           0.0          0.0       0      0  0
 75         0.0          0.0          0.0           0.0           0.0          0.0       0      0  0
 76         0.0          0.0          0.0           0.0           0.0          0.0       0      0  0
 77         0.0          0.0          0.0           0.0           0.0          0.0       0      0  0
 78         0.0          0.0          0.0           0.0           0.0          0.0       0      0  0
 79         0.0          0.0          0.0           0.0           0.0          0.0       0      0  0
 80         0.0          0.0          0.0           0.0           0.0          0.0       0      0  0
```

```
COLUMN      8            9           10          11            12
IDENT   FACES         LINE END POINTS          INTERSECTING SURFACES
   1     36    1         1   -2       -1   49       1   32      32   19
   2      4   37         2   -3      -25   51       1   32      20   21
   3      4   41         3   -4       19  -52       1   31      21   22
   4      4   45         4   -5      -18   53       1   30     -22   23
   5      4   49         5   -6       17  -54       1   29      23   24
   6     -4   53         6  -63      -16   55       1   28     -24   25
   7     -4   57        -7    8       -9   56       1   27      25   26
   8     -4   61        -8    9       -8   57       1   26      26   27
   9     -4   65        -9   10       -7   58       1    1     -27   28
  10     -4   69        64  -11        6  -59       1    1      28   29
  11     -5   73       -10   12       -5   60       1    1      29   30
  12      4   78        65  -13       -4   61       1    1     -30   31
  13      4   82       -12   14       -3   62       1    1      31   32
  14     -4   86        66  -15        0    0       1    1       0    0
  15     -4   90        -8   14      -16    0       1    1       1    1
  16      4   94        67  -16      -40    0       1   25      28    2
  17    -20   98        16  -17      -25    0       1   24       4    2
  18     10  118        17  -18      -70    0       1   23       3    2
  19     -4  128        18  -19      -44    0       1   22       3    2
  20     -4  132        19  -20        0    0       1   21      12    2
  21     -5  136        20  -21        0    0       1    1       0    0
  22      4  141       -21   22        0    0      -1    9       0    0
  23     -4  145        22  -23        0    0      -1    2       0    0
  24      4  149        23  -24        0    0      -1    5       0    0
  25     -4  153       -21   24        0    0      -7    1       0    0
  26      4  157        20  -25        0    0       1   21       0    0
  27      4  161        25  -26        0    0       1   20       0    0
  28      4  165         1  -26        0    0      19    1       0    0
  29     -4  169        22  -27        0    0       2    9       0    0
  30     -4  173        27  -28        0    0      -2   18       0    0
  31     -4  177        23  -28        0    0       5    2       0    0
  32     -6  181        29  -30        0    0       3   18       0    0
  33      0    0        30  -31        0    0       3    8       0    0
  34      0    0        31  -32        0    0       3   10       0    0
  35      0    0        29  -32        0    0       4    3       0    0
  36      0    0        29  -33        0    0      18    4       0    0
  37      0    0        32  -34        0    0       4   10       0    0
  38      0    0        33  -34        0    0       6    4       0    0
  39      0    0        28  -35        0    0      -5   18       0    0
  40      0    0        24  -35        0    0       7    5       0    0
  41      0    0       -34   36        0    0       6   10       0    0
  42      0    0       -36   37        0    0       6    8       0    0
  43      0    0        33  -37        0    0      18    6       0    0
  44      0    0       -35   38        0    0      -7   18       0    0
  45      0    0       -21   38        0    0       9    7       0    0
  46      0    0        30  -37        0    0       8   18       0    0
  47      0    0       -31   36        0    0      10    8       0    0
  48      0    0       -27   38        0    0     -18    9       0    0
  49      0    0       -39   40        0    0      11   15       0    0
  50      0    0        40  -68        0    0      11   16       0    0
  51      0    0       -41   42        0    0      11   12       0    0
  52      0    0        69  -43        0    0      11   13       0    0
  53      0    0       -39   43        0    0      14   11       0    0
  54      0    0       -42   44        0    0      13   12       0    0
  55      0    0       -41   45        0    0      12   16       0    0
  56      0    0       -44   45        0    0     -17   12       0    0
  57      0    0        43  -46        0    0      14   13       0    0
  58      0    0       -44   46        0    0     -13   17       0    0
  59      0    0       -39   47        0    0      15   14       0    0
  60      0    0       -46   47        0    0     -14   17       0    0
  61      0    0       -47   48        0    0     -15   17       0    0
  62      0    0        40  -48        0    0      16   15       0    0
  63      0    0       -45   48        0    0     -17   16       0    0
  64      0    0       -47   49        0    0      17   32       0    0
```

```
COLUMN      8             9           10              11            12
IDENT   FACES          LINE END POINTS          INTERSECTING SURFACES
 65        0    0      -49   50      0     0       17   19        0     0
 66        0    0      -50   51      0     0       17   20        0     0
 67        0    0      -51   52      0     0       17   21        0     0
 68        0    0       52  -53      0     0       17   22        0     0
 69        0    0      -53   54      0     0       17   23        0     0
 70        0    0       54  -55      0     0       17   24        0     0
 71        0    0      -55   56      0     0       17   25        0     0
 72        0    0      -56   57      0     0       17   26        0     0
 73        0    0      -57   58      0     0       17   27        0     0
 74        0    0       70  -59      0     0       17   28        0     0
 75        0    0      -59   60      0     0       17   29        0     0
 76        0    0      -60   61      0     0       17   30        0     0
 77        0    0      -61   62      0     0       17   31        0     0
 78        0    0      -47   62      0     0       32   17        0     0
 79        0    0       37  -38      0     0       18   18        0     0
 80        0    0      -26   50      0     0      -19   20        0     0
```

```
                              LSTFAC

   1     2     3     4     5          6     7     8     9    10
  11    10    12    13    12         14    15    14     8    14
  12    10     9    16    17         18    19    20    21    22
  23    24    21    20    25         26    23    22    27    28
  29    30    31    32    33         29    32    34    24    23

  28    35    33    34    36         37    24    35    38    21
  30    37    36    31    22         21    38    27    34    32
  31    36    39    40    41         42    43    44    42    41
  45    46    43    42    44         47    39    43    46    39
  47    48    40    40    48         45    41    47    46    44

  45    48    47    49    50         51    52    53    54    55
  56    57    58    59    60         61    62    38    35    28
  27    38    37    30    29         33    37    26    50    49
   1    25    51    50    26         19    52    51    25    20
  18    53    52    19    17         54    53    18    16    55

  54    17     9    56    55         16     8    57    56     9
   7    58    57     8     6         59    58     7     5    60
  59     6     4    61    60          5     3    62    61     4
   3     2     1    49    47         62

                              LSTBFC

   2     2     2     2     1          1     1     1     1     1
   1     1     2
```

```
                          VISIBLE LINES
    XSTART        YSTART                  XEND         YEND
  0.677631     -0.029059              0.528135    -0.028813
  0.677631     -0.029059              0.832970    -0.042945
  0.528135     -0.028813              0.381145    -0.028572
  0.381145     -0.028572              0.202804    -0.044989
  0.202804     -0.044989             -0.137045    -0.044973
 -0.137045     -0.044973             -1.000000    -0.103245
  1.000000     -0.290405              0.203018    -0.044959
 -1.000000     -0.851212              0.001951    -0.044846
 -1.000000     -0.185074             -0.245694    -0.052309
  1.000000     -0.046789              0.832970    -0.042945
```

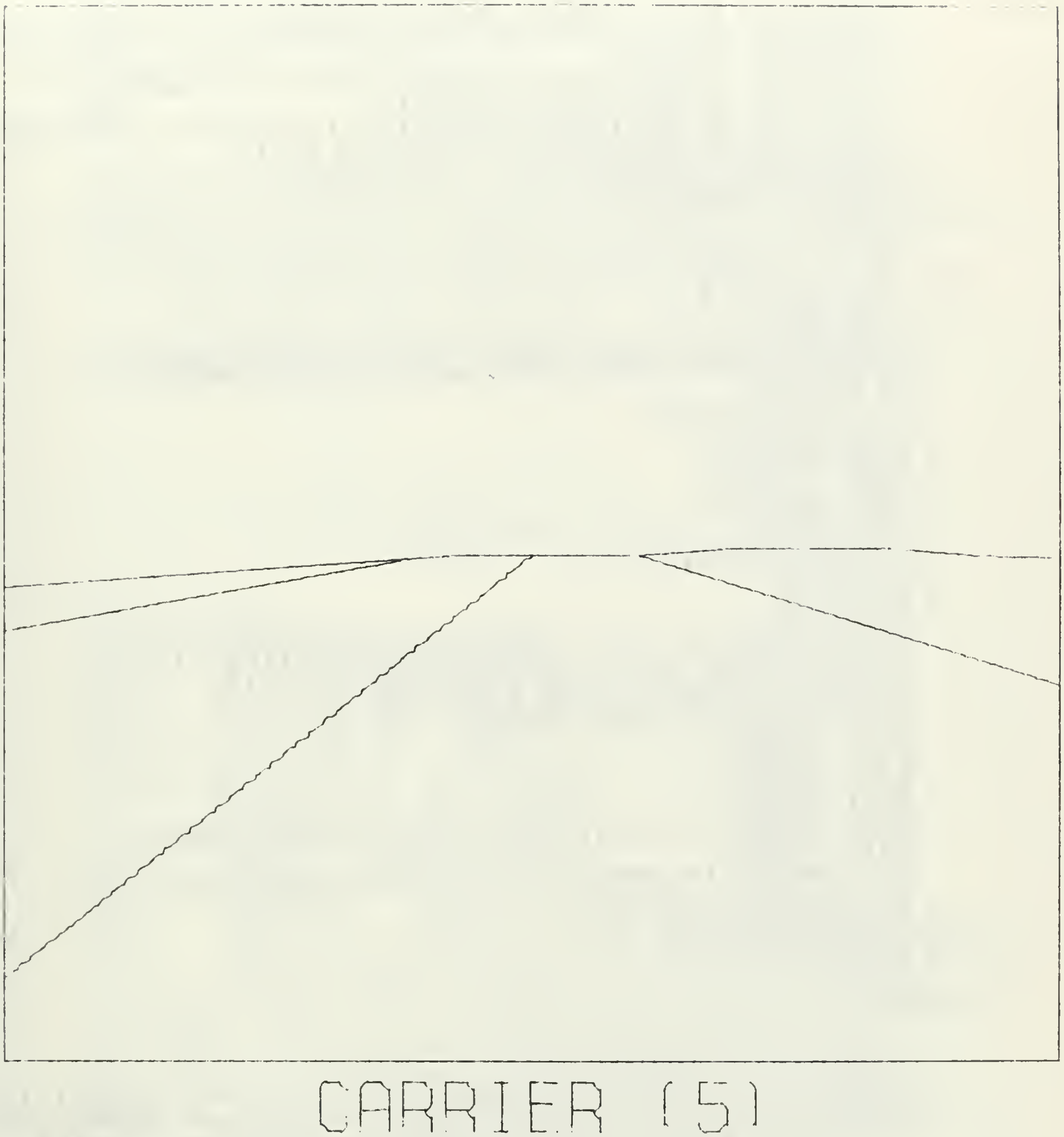


FIGURE 39

COMPUTER PROGRAM FOR THE IBM 360-67

```
      COMMON/SDFB/ALPHA,BETA,GAMMA,PSI,THETA,PHI,OBJMV(3),
     1VIEWMV(3),FOCUS,YOFF,ZOFF,SCLE,HMATRX(4,4),OBSPOS(12)
      INTEGER*2 IARRAY
      DIMENSION ARRAY(80,13),IARRAY(2,80,13),FNORML(80,3)
      DIMENSION LSTFAC(250),LSTBFC(1)
      DIMENSION XSTART(100),YSTART(100),XEND(100),YEND(100)
      EQUIVALENCE (ARRAY(1,1),IARRAY(1,1,1))
      EQUIVALENCE (FNORML(1,3),LSTBFC(1))
      EXTERNAL READCR
      IRPT=6
      N=80
      DO 11 I=1,N
      DO 11 J=1,13
      ARRAY(I,J)=0.0
   11 CONTINUE
      CALL DESCRB(ARRAY,IARRAY,FNORML,LSTFAC,N,READCR,
     1XSTART,YSTART,XEND,YEND,NLINES,LSTBFC,NUSED)
      NFACES=32
      DO 1 J=1,NFACES
      CALL FACES
    1 CONTINUE
      CALL LINES
      CALL CCVCVX
    2 CONTINUE
      IRPT=IRPT-1
      IF(IRPT.GT.0)GO TO 3
      GO TO 9
    3 CONTINUE
      CALL ERASE
      READ(5,5)ALPHA,BETA,GAMMA,(OBJMV(I),I=1,3)
      READ(5,5)PSI,THETA,PHI,(VIEWMV(J),J=1,3)
      READ(5,10)FOCUS,SCLE,  YOFF,ZOFF
      CALL CONV3D(N,ARRAY,NUSED)
      CALL HDNSRF
      CALL ALTER
      CALL CMPUT
      CALL DATSTR
      CALL DRAWVP(NLINES,XSTART,YSTART,XEND,YEND)
      CALL LABEL
      CALL DRAWIT(NLINES,XSTART,YSTART,XEND,YEND,IRPT)
      GO TO 2
    5 FORMAT(6F10.3)
   10 FORMAT(4F10.6)
    9 CONTINUE
      STOP
      END


      SUBROUTINE CONV3D(N,PNTLST,NUSED)
      COMMON/SDFB/ALPHA,BETA,GAMMA,PSI,THETA,PHI,OBJMV(3),
     1VIEWMV(3),FOCUS,YOFF,ZOFF,SCLE,HMATRX(4,4),OBSPOS(12)
      DIMENSION PNTLST(N,13),DISPLY(3),HCRD3D(4),TEMP(4)
C          THIS PROGRAM USES THE FOURTH COORDINATE OF THE
C          HOMOGENEOUS SET EQUAL TO 1.0 AND THEREFORE REDUCES
C          THE NUMBER OF MULTIPLICATIONS NECESSARY
      CALL VMVMAT
      TEMP(4)=1.0
      DO 5 I=1,NUSED
      DO 1 IN=1,4
    1 HCRD3D(IN)=0.0
C          CONVERT THREE DIMENSIONAL COORDINATES TO FOUR
C    1DIMENSIONAL HOMOGENOUS COORDINATES
      DO 2 IM=1,3
    2 TEMP(IM)=PNTLST(I,IM)
      DO 3 K=1,4
      DO 3 J=1,4
      HCRD3D(K)=HCRD3D(K)+TEMP(J)*HMATRX(J,K)
    3 CONTINUE
      DO 4 ID=2,3
      DISPLY(ID)=HCRD3D(ID)/HCRD3D(4)
    4 CONTINUE
C     SAVE DEPTH INFORMATION
      DISPLY(1)=HCRD3D(1)
      DO 5 IS=1,3
    5 PNTLST(I,IS+3)=DISPLY(IS)
      RETURN
      END


      SUBROUTINE VMVMAT
      COMMON/SDFB/ALPHA,BETA,GAMMA,PSI,THETA,PHI,OBJMV(3),
     1VIEWMV(3),FOCUS,YOFF,ZOFF,SCLE,HMATRX(4,4),OBSPOS(12)
      DIMENSION ROTMAT(4,4),ORIMAT(4,4),TEMP(3)
      DIMENSION TEMPY(3),TEMPZ(3),TEMPF(3)
C         THIS SUBROUTINE CALCULATES THE 4 X 4 MATRIX FOR
C         PERSPCTIVE TRANSFORMATIONS
C
C                                               .W
C                                .Z             .
C                                .              .
C                                .        . . . .
C                                .        U      .
C                                .        OBJECT  .
C                    . . . . . . .        AXES      .V
C                    X             .
C                       VIEWPLANE    .
C                         AXES         .
C                                        .Y
C
C     DEFINITIONS
C     ANGLES (ENTERED IN DEGREES)
C     ALPHA  ROTATION OF U INTO V
C     BETA   ROTATION OF W INTO U
C     GAMMA  ROTATION OF V INTO W
C     PSI    ROTATION OF X INTO Y
C     THETA  ROTATION OF Z INTO X
C     PHI    ROTATION OF Y INTO Z
C     OBJMV  MOVEMENT OF UVW ORIGIN
C            MEASURED IN XYZ COORDS
C     VIEWMV MOVEMENT OF XYZ AXES
C     FOCUS  VIEWING DISTANCE FROM
C            PICTURE PLANE (YZ PLANE)
C     YOFF   HORIZONTAL OFFSET IN PICTURE PLANE
C     ZOFF   VERTICAL   OFFSET IN PICTURE PLANE
C     SCLE   NORMALIZED DISTANCE FROM CENTER TO EDGE OF PICT
C            NORMAL VALUE FOR FULL PICTURE IS 1/2
C     NORMAL RATIO OF SCLE TO FOCUS IS BETWEEN 1/4 AND 1/3
C     IF FOCAL LENGTH IS INCREASED WITH SCLE HELD CONSTANT
C     THE RESULT IS AS IF A TELEPHOTO LENS WERE USED
C     DECREASING THE FOCAL LENGTH CORRESPONDS TO USING A
C     WIDE ANGLE LENS
C     VIEWING ANGLES MEASURED VERTICALLY AND HORIZONTALLY
C     FROM CENTER ARE APPROXIMATELY 14 DEGREES FOR A RATIO
C     OF 1/4 AND 18.5 DEGREES FOR A RATIO OF 1/3
C
      CALL ROTATN(ROTMAT,ALPHA,BETA,GAMMA)
      DO 1 I=1,3
    1 ROTMAT(4,I)=OBJMV(I)-VIEWMV(I)
      A=-PSI
      B=-THETA
      C=-PHI
      CALL FXRTAT(ORIMAT,A,B,C)
      CALL PROD(ROTMAT,ORIMAT,4,4,4,HMATRX,4,4,4)
      FCTR1=-SCLE/FOCUS
      HMATRX(1,4)=HMATRX(1,1)*FCTR1
      HMATRX(2,4)=HMATRX(2,1)*FCTR1
      HMATRX(3,4)=HMATRX(3,1)*FCTR1
      HMATRX(4,4)=(HMATRX(4,1)-FOCUS)*FCTR1
      IF(YOFF.EQ.0.)GO TO 3
      FCTR2=YOFF/SCLE
      DO 2 I=1,4
    2 HMATRX(I,2)=HMATRX(I,2)+FCTR2*HMATRX(I,4)
    3 CONTINUE
      IF(ZOFF.EQ.0.)GO TO 5
      FCTR3=ZOFF/SCLE
      DO 4 I=1,4
    4 HMATRX(I,3)=HMATRX(I,3)+FCTR3*HMATRX(I,4)
    5 CONTINUE
      CALL ROTATN(ORIMAT,PSI,THETA,PHI)
      DO 6 I=1,3
      TEMP(I)=VIEWMV(I)-OBJMV(I)
      TEMPF(I)=ORIMAT(1,I)*FOCUS+TEMP(I)
      TEMPY(I)=ORIMAT(2,I)+TEMP(I)
    6 TEMPZ(I)=ORIMAT(3,I)+TEMP(I)
      A=-ALPHA
      B=-BETA
      C=-GAMMA
      CALL ROTATN(ROTMAT,A,B,C)
      DO 7 I=1,12
    7 OBSPOS(I)=0.
      DO 8 J=1,3
      JJ=J+3
      JJJ=J+6
      J4J=J+9
      DO 8 I=1,3
      OBSPOS(JJ)=OBSPOS(JJ)+TEMPY(I)*ROTMAT(I,J)
      OBSPOS(JJJ)=OBSPOS(JJJ)+TEMPZ(I)*ROTMAT(I,J)
      OBSPOS(J4J)=OBSPOS(J4J)+TEMPF(I)*ROTMAT(I,J)
    8 OBSPOS(J)=OBSPOS(J)+TEMP(I)*ROTMAT(I,J)
      RETURN
      END


      SUBROUTINE FXRTAT(RMATRX,A,B,C)
      DIMENSION RMATRX(4,4)
C            THIS SUBROUTINE CALCULATES THE 4 X 4 ROTATION
C            MATRIX FOR FIXED ANGLE ROTATION.
C            CALCULATE DIRECTIONAL COSINES
C            AND PLACE IN ROTATION MATRIX
      TPI=3.1416/180.
      ALPHA=TPI*A
      BETA=TPI*B
      GAMMA=TPI*C
      SALPHA=SIN(ALPHA)
      CALPHA=COS(ALPHA)
      SBETA=SIN(BETA)
      CBETA=COS(BETA)
      SGAMMA=SIN(GAMMA)
      CGAMMA=COS(GAMMA)
      TEMP1=CGAMMA*SALPHA
      TEMP2=SGAMMA*CALPHA
      TEMP3=SGAMMA*SALPHA
      TEMP4=CGAMMA*CALPHA
      RMATRX(1,1)=CBETA*CALPHA
      RMATRX(1,2)=TEMP1+TEMP2*SBETA
      RMATRX(1,3)=TEMP3-TEMP4*SBETA
      RMATRX(2,1)=-CBETA*SALPHA
      RMATRX(2,2)=TEMP4-TEMP3*SBETA
      RMATRX(2,3)=TEMP2+TEMP1*SBETA
      RMATRX(3,1)=SBETA
      RMATRX(3,2)=-CBETA*SGAMMA
      RMATRX(3,3)=CGAMMA*CBETA
      DO 1 I=1,3
      RMATRX(I,4)=0.
    1 RMATRX(4,I)=0.
      RMATRX(4,4)=1.
      RETURN
      END
      SUBROUTINE ROTATN(RMATRX,A,B,C)
      DIMENSION RMATRX(4,4)
C           THIS SUBROUTINE CALCULATES THE 4 X 4 ROTATION
C           MATRIX FOR EULER ANGLE ROTATION
C           CALCULATE DIRECTIONAL COSINES
C           AND PLACE IN ROTATION MATRIX
      TPI=3.1416/180.
      ALPHA=TPI*A
      BETA=TPI*B
      GAMMA=TPI*C
      SALPHA=SIN(ALPHA)
      CALPHA=COS(ALPHA)
      SBETA=SIN(BETA)
      CBETA=COS(BETA)
      SGAMMA=SIN(GAMMA)
      CGAMMA=COS(GAMMA)
      TEMP1=CGAMMA*SALPHA
      TEMP2=SGAMMA*CALPHA
      TEMP3=SGAMMA*SALPHA
      TEMP4=CGAMMA*CALPHA
      RMATRX(1,1)=CBETA*CALPHA
      RMATRX(1,2)=CBETA*SALPHA
      RMATRX(1,3)=-SBETA
      RMATRX(2,1)=TEMP2*SBETA-TEMP1
      RMATRX(2,2)=TEMP3*SBETA+TEMP4
      RMATRX(2,3)=SGAMMA*CBETA
      RMATRX(3,1)=SBETA*TEMP4+TEMP3
      RMATRX(3,2)=SBETA*TEMP1-TEMP2
      RMATRX(3,3)=CBETA*CGAMMA
      DO 1 I=1,3
      RMATRX(I,4)=0.
    1 RMATRX(4,I)=0.
      RMATRX(4,4)=1.
      RETURN
      END


      SUBROUTINE PROD (A,B,N,M,L,C,ND,MD,LD)
      DIMENSION A(ND,MD),B(MD,LD),C(ND,LD)
      DO 1 I=1,ND
      DO 1 J=1,LD
    1 C(I,J)=0.
      DO 151 I=1,N
      DO 151 J=1,L
      DO 151 K=1,M
  151 C(I,J) = C(I,J) + A(I,K)*B(K,J)
      RETURN
      END


      SUBROUTINE DESCRB(ARRAY,IARRAY,FNORML,LSTFAC,N,SUBR1,
     1XSTART,YSTART,XEND,YEND,/JDSPLY/,LSTBFC,/NPNTR/)
      COMMON/SDFB/ALPHA,BETA,GAMMA,PSI,THETA,PHI,OBJMV(3),
     1VIEWMV(3),FOCUS,YOFF,ZOFF,SCLE,HMATRX(4,4),OBSPOS(12)
      COMMON /DSCRB/LPNTR,JPNTR,IFACE,TEMPIN(9),ICROSS,
     1INRSCT,ICOUNT,NTEMP,CURSEG,LLOW,LHIGH,XLOW,XHIGH,YLOW,
     2YHIGH,XNEAR,YNEAR,DPTHNR, AA(3),BB(3),CC(3),PP(3),AAL,
     3BBL,LRESTR
      INTEGER*2 IARRAY(2,N,13)
      DIMENSION ARRAY(N,13),LSTFAC(1),LSTBFC(1)
      DIMENSION FNORML(N,3)
      DIMENSION IWRKNR(20),JGSET(20),INTLVR(20)
      DIMENSION XSTART(1),YSTART(1),XEND(1),YEND(1)
      INTEGER POINT,CURSEG,TPNTR,TEMP
      TRISNS(A0,B0,A1,B1,A2,B2,CONST)=A0*B1+A2*B0-A0*B2-
     1A1*B0+CONST
C
C           N IS THE NUMBER OF DIFFERENT POINTS EXPECTED
```

```
C         SUBR1 IS THE SUBROUTINE THAT WILL ENTER DATA
C         CALLING NAME OF SUBR1 MUST BE DECLARED EXTERNAL
C         IN THE MAIN CALLING PROGRAM IN ORDER THAT ITS
C         NAME WILL BE RECOGNIZED AS A SUBROUTINE
C         DATA IS ENTERED IN THE FIRST THREE LOCATIONS
C         OF TEMPIN
C         SUBR1 SHOULD READ IN NUMBER OF POINTS TO THE FACE
C         THIS SHOULD BE FOLLOWED BY THE THREE COORDINATES
C         OF EACH POINT, ONE POINT AT A TIME, INTO ARRAY
C         TEMPIN
C         IF DESIRED, READCR MAY BE USED TO ENTER DATA
C         FORMAT EXPECTED IS I110 FOR THE NUMBER OF POINTS
C         OF THAT FACE TO FOLLOW
C         SUBSEQUENT CARDS WILL USE THE FORMAT 3F20.6 FOR
C         COORDINATES X,Y,Z
C        THE FIRST POINT ENTERED MUST BE A CORNER POINT
C         I.E., THE LAST, FIRST, AND SECOND POINTS MUST
C         NOT BE COLLINEAR
C
      NPNTR=0
      LRESTR=0
      LPNTR=0
      IFPNTR=0
      LSPNTR=0
      JDSPLY=0
      JPNTR=0
      CALL QSCRB1(N,ARRAY,IARRAY,FNORML,LSTFAC,LSTBFC,
     1IWRKNR,INTLVR,JGSET,XSTART,YSTART,XEND,YEND,JDSPLY)
      DO 1 I=1,N
      ARRAY(I,7)=0.
      ARRAY(I,13)=0.
    1 CONTINUE
      RETURN
C
      ENTRY ERASE
      DO 7 I=1,N
    7 ARRAY(I,13)=0.
      IF(LRESTR.NE.0)GO TO 30
      DO 6 I=1,LPNTR
      TEMP=IARRAY(2,I,9)
    6 IARRAY(2,I,9)=IABS(TEMP)
      RETURN
C
      ENTRY FACES
      NPT=1
      CALL SUBR1(NPT,TEMPIN)
      IFPNTR=IFPNTR+1
      IARRAY(1,IFPNTR,8)=NPT
      IARRAY(2,IFPNTR,8)=LSPNTR+1
      DO 13 IN=1,NPT
      CALL SUBR1(0,TEMPIN)
   21 DO 22 I=1,NPNTR
      IF(TEMPIN(1).EQ.ARRAY(I,1).AND.TEMPIN(2).EQ.ARRAY(I,2)
     1.AND.TEMPIN(3).EQ.ARRAY(I,3))GO TO 14
   22 CONTINUE
      NPNTR=NPNTR+1
      NEW=NPNTR
      DO 11 I=1,3
   11 ARRAY(NPNTR,I)=TEMPIN(I)
   12 LSPNTR=LSPNTR+1
   13 LSTFAC(LSPNTR)=NEW
      RETURN
   14 NEW=I
      GO TO 12
C
      ENTRY LINES
   30 CONTINUE
      DO 40 I=1,IFPNTR
      IR=1
      NBR=IARRAY(1,I,8)
      NBR=IABS(NBR)
      TPNTR=IARRAY(2,I,8)
      LSTRT=TPNTR
      LPNT=LSTFAC(TPNTR)
   31 TPNTR=TPNTR+1
      NPNT=LSTFAC(TPNTR)
   32 IR=IR+1
      IF(NPNT.LT.LPNT)GO TO 38
      IB=1
      LLOW=LPNT
      LHIGH=NPNT
   39 CONTINUE
      DO 16 J=1,LPNTR
      IF(IARRAY(1,J,9).EQ.LLOW.AND.IARRAY(2,J,9).EQ.LHIGH)GO
     1TO 37
   16 CONTINUE
      LPNTR=LPNTR+1
      IF(LRESTR.NE.0)GO TO 33
   17 IARRAY(IB,LPNTR,11)=I
   33 IARRAY(1,LPNTR,9)=LLOW
      IARRAY(2,LPNTR,9)=LHIGH
   34 LPNT=NPNT
      IF(IR-NBR)31,36,40
   36 NPNT=LSTFAC(LSTRT)
      GO TO 32
   37 IARRAY(IB,J,11)=I
      GO TO 34
   38 LLOW=NPNT
      LHIGH=LPNT
      IB=2
      GO TO 39
   40 CONTINUE
      LRESTR=0
      RETURN
C
      ENTRY CCVCVX
      DO 49 I=1,IFPNTR
      NBR=IARRAY(1,I,8)
      NBR=IABS(NBR)
      TPNTR=IARRAY(2,I,8)
      POINT=LSTFAC(TPNTR+NBR-1)
      DO 41 J=1,3
   41 TEMPIN(J)=ARRAY(POINT,J)
      POINT=LSTFAC(TPNTR)
      DO 42 J=1,3
   42 TEMPIN(J+3)=ARRAY(POINT,J)
      POINT=LSTFAC(TPNTR+1)
      DO 43 J=1,3
   43 TEMPIN(J+6)=ARRAY(POINT,J)
      A=SQRT((TEMPIN(1)-TEMPIN(4))**2+(TEMPIN(2)-TEMPIN(5))
     1**2+(TEMPIN(3)-TEMPIN(6))**2)
      B=SQRT((TEMPIN(7)-TEMPIN(4))**2+(TEMPIN(8)-TEMPIN(5))
     1**2+(TEMPIN(9)-TEMPIN(6))**2)
      ABI=1./(A*B)
      FNORML(I,1)=ABI*((TEMPIN(2)-TEMPIN(5))*(TEMPIN(9)-
     1TEMPIN(6))-(TEMPIN(3)-TEMPIN(6))*(TEMPIN(8)-
     2TEMPIN(5)))
      FNORML(I,2)=ABI*((TEMPIN(3)-TEMPIN(6))*(TEMPIN(7)-
     1TEMPIN(4))-(TEMPIN(1)-TEMPIN(4))*(TEMPIN(9)-
     2TEMPIN(6)))
      FNORML(I,3)=ABI*((TEMPIN(1)-TEMPIN(4))*(TEMPIN(8)-
     1TEMPIN(5))-(TEMPIN(2)-TEMPIN(5))*(TEMPIN(7)-
     2TEMPIN(4)))
   49 CONTINUE
      DO 47 I=1,LPNTR
      IFACE1=IARRAY(1,I,11)
      IFACE1=IABS(IFACE1)
      IFACE2=IARRAY(2,I,11)
      LLOW=IARRAY(1,I,9)
      LHIGH=IARRAY(2,I,9)
      DO 44 J=1,3
      AA(J)=FNORML(IFACE1,J)
   44 BB(J)=FNORML(IFACE2,J)
      CALL CROSS(AA,BB,CC)
      DO 45 J=1,3
   45 AA(J)=ARRAY(LHIGH,J)-ARRAY(LLOW,J)
      SCALAR=0.
      DO 46 J=1,3
   46 SCALAR=AA(J)*CC(J)+SCALAR
      IF(SCALAR.GE.0.)GO TO 47
      IARRAY(1,I,11)=-IFACE1
      IARRAY(1,LLOW,7)=1
      IARRAY(1,LHIGH,7)=1
   47 CONTINUE
      RETURN
C
      ENTRY HDNSRF
      JPNTR=0
      DO 54 I=1,IFPNTR
      NBR=IARRAY(1,I,8)
      NBR=IABS(NBR)
      TPNTR=IARRAY(2,I,8)
      POINT=LSTFAC(TPNTR+NBR-1)
      DO 51 J=1,2
   51 TEMPIN(J)=ARRAY(POINT,J+4)
      POINT=LSTFAC(TPNTR)
      DO 52 J=3,4
   52 TEMPIN(J)=ARRAY(POINT,J+2)
      POINT=LSTFAC(TPNTR+1)
      DO 53 J=5,6
   53 TEMPIN(J)=ARRAY(POINT,J)
      DEPTHN=(TEMPIN(1)-TEMPIN(3))*(TEMPIN(6)-TEMPIN(4))-
     1(TEMPIN(5)-TEMPIN(3))*(TEMPIN(2)-TEMPIN(4))
      IF(DEPTHN.GE.0.)GO TO 55
      IARRAY(1,I,8)=NBR
   54 CONTINUE
      DO 58 I=1,LPNTR
      IFLAG1=0
      TEMP=IARRAY(1,I,9)
      TEMP=IABS(TEMP)
      IFACE1=IARRAY(1,I,11)
      IF(IFACE1.GT.0)GO TO 57
      IFACE1=-IFACE1
      IFLAG1=1
   57 CONTINUE
      IFACE2=IARRAY(2,I,11)
      IF(IARRAY(1,IFACE1,8).LT.0)IFLAG1=IFLAG1+1
      IF(IARRAY(1,IFACE2,8).LT.0)IFLAG1=IFLAG1+1
      IF(IFLAG1.GE.2)GO TO 56
      IARRAY(1,I,9)=TEMP
   58 CONTINUE
      RETURN
   55 IARRAY(1,I,8)=-NBR
      GO TO 54
   56 IARRAY(1,I,9)=-TEMP
      GO TO 58
C
      ENTRY CMPUT
C            EXAMINE SEGMENTS ORIGINATING AT SPECIFIED NODE
      JDSPLY=0
      DO 104 ILK=1,LPNTR
      LLOW=IARRAY(1,ILK,9)
      IF(LLOW.LT.0)GO TO 104
      XLOW=ARRAY(LLOW,5)
      YLOW=ARRAY(LLOW,6)
      IF(IARRAY(2,LLOW,13).NE.0)GO TO 105
      CALL VRTNTR
  105 CONTINUE
C            SEARCH FIRST NODE OF ALL SEGMENTS
      DO 114 IL=ILK,LPNTR
      IF(IARRAY(1,IL,9).NE.LLOW)GO TO 114
      LHIGH=IARRAY(2,IL,9)
      IF(LHIGH.LT.0)GO TO 114
C            ENTER NATURE AT INITIAL VERTEX
  112 NBR=IARRAY(1,LLOW,13)
      TPNTR=IARRAY(2,LLOW,13)
      IWRKNR(1)=NBR
      IF(NBR.EQ.0)GO TO 301
      IDRAW=0
  107 IEND=NBR+1
      DO 111 J=2,IEND
  111 IWRKNR(J)=LSTBFC(TPNTR+J-2)
  113 CONTINUE
C         FIND DIRECTION NUMBERS
      XHIGH=ARRAY(LHIGH,5)
      YHIGH=ARRAY(LHIGH,6)
      AAL=XHIGH-XLOW
      BBL=YHIGH-YLOW
C         FIND LOSING INTERSECTIONS
      CURSEG=IL
      CALL LNTRSC
      IF(INRSCT.EQ.0)GO TO 103
C         SEARCH FOR INTERSECTION AT INITIAL VERTEX
      ICOUNT=INRSCT
      IJK=LPNTR+1
      NTEMP=0
      DO 109 I=1,INRSCT
      IJK=IJK+1
      IF(IARRAY(1,IJK,9).LT.0)GO TO 109
      IF(XLOW.NE.FNORML(I,1).OR .YLOW.NE.FNORML(I,2))GO TO
     1109
C         INTERSECTION OCCURS AT INITIAL VERTEX
      NTEMP=NTEMP+1
      INTLVR(NTEMP)=IARRAY(1,IJK,9)
      IARRAY(1,IJK,9)=-IARRAY(1,IJK,9)
  109 CONTINUE
      IF(NTEMP.EQ.0)GO TO 151
      XNEAR=XLOW+.0001*AAL
      YNEAR=YLOW+.0001*BBL
C         COMPUTE DEPTH OF POINT NEAR VERTEX
      TEMPIN(1)=.001*ARRAY(LHIGH,1)-ARRAY(LLOW,1)*.999
      TEMPIN(2)=.001*ARRAY(LHIGH,2)-ARRAY(LLOW,2)*.999
      TEMPIN(3)=.001*ARRAY(LHIGH,3)-ARRAY(LLOW,3)*.999
      DPTHNR=DEPTH(TEMPIN(1),TEMPIN(2),TEMPIN(3))
      CALL TSTNNP
      IF(IDRAW.EQ.1.AND.IWRKNR(1).NE.0)GO TO 302
      IF(IDRAW.EQ.0.AND.IWRKNR(1).EQ.0)GO TO 303
      GO TO 151
  301 JDSPLY=JDSPLY+1
      XSTART(JDSPLY)=XLOW
      YSTART(JDSPLY)=YLOW
      IDRAW=1
      GO TO 113
  302 JDSPLY=JDSPLY-1
      IDRAW=0
      GO TO 151
  303 JDSPLY=JDSPLY+1
      XSTART(JDSPLY)=XLOW
      YSTART(JDSPLY)=YLOW
      IDRAW=1
  151 CONTINUE
      IF(ICOUNT.EQ.0) GO TO 103
      NTEMP=0
      IJK=LPNTR+1
      DO 152 I=1,INRSCT
      IJK=IJK+1
      IJKTST=IARRAY(1,IJK,9)
      IF(IJKTST.LT.0)GO TO 152
      IF(XHIGH.EQ.FNORML(I,1).AND.YHIGH.EQ.FNORML(I,2))GO
     1TO 153
  152 CONTINUE
      IFLAG=0
      IF(ICOUNT.LE.NTEMP)GO TO 201
      GO TO 154
  153 NTEMP=NTEMP+1
      INTLVR(NTEMP)=IJKTST
      IARRAY(1,IJK,9)=-IJKTST
      GO TO 152
  154 CONTINUE
      IJK=LPNTR+1
      RLNGTH=1.E6
      DO 155 I=1,INRSCT
      IJK=IJK+1
      IJKTST=IARRAY(1,IJK,9)
      IF(IJKTST.LT.0)GO TO 155
      ALSEG=FNORML(I,1)-XLOW
      BLSEG=FNORML(I,2)-YLOW
      TLNGTH=ABS(ALSEG)+ABS(BLSEG)
      IF(TLNGTH-RLNGTH)156,157,155
  156 RLNGTH=TLNGTH
      ISEGJ=IJKTST
      NXTINT=IJK
      MULTPL=1
      NI=I
      GO TO 155
  157 IF(IFLAG.EQ.0)MULTPL=MULTPL+1
  155 CONTINUE
      IFACE1=IARRAY(1,ISEGJ,11)
      IFACE1=IABS(IFACE1)
      IFACE2=IARRAY(2,ISEGJ,11)
      IF(MULTPL.GT.1)GO TO 162
  165 CONTINUE
      ITSTNR=IWRKNR(1)
      IF(IARRAY(1,IFACE1,8).LT.0)GO TO 160
      CALL DELETE(IWRKNR,IFACE1)
      IF(ITSTNR.NE.IWRKNR(1))GO TO 159
      IWRKNR(1)=IWRKNR(1)+1
      IWRKNR(ITSTNR+2)=IFACE1
      CALL DELETE(IWRKNR,IFACE2)
      GO TO 161
  159 IF(IARRAY(1,IFACE2,8).LT.0)GO TO 161
      IWRKNR(ITSTNR+1)=IFACE2
      IWRKNR(1)=IWRKNR(1)+1
      GO TO 161
  160 CALL DELETE(IWRKNR,IFACE2)
      IF(ITSTNR.NE.IWRKNR(1))GO TO 161
      IWRKNR(1)=IWRKNR(1)+1
      IWRKNR(ITSTNR+2)=IFACE2
  161 IARRAY(1,NXTINT,9)=-ISEGJ
      IF(IDRAW.EQ.1.AND.IWRKNR(1).NE.0)GO TO 304
      IF(IDRAW.EQ.0.AND.IWRKNR(1).EQ.0)GO TO 305
  168 ICOUNT=ICOUNT-1
      IF(ICOUNT.LE.NTEMP)GO TO 201
      MULTPL=MULTPL-1
      IF(MULTPL.LE.0)IFLAG=0
      GO TO 154
  304 XEND(JDSPLY)=FNORML(NI,1)
      YEND(JDSPLY)=FNORML(NI,2)
      CALL WINDOW
      IDRAW=0
      GO TO 168
  305 JDSPLY=JDSPLY+1
      XSTART(JDSPLY)=FNORML(NI,1)
      YSTART(JDSPLY)=FNORML(NI,2)
      IDRAW=1
      GO TO 168
  162 CONTINUE
C            MULTIPLE INTERSECTIONS
      IRPT=1
      ISCTND=IARRAY(1,ISEGJ,9)
      XINSCT=FNORML(NI,1)
      YINSCT=FNORML(NI,2)
  163 XSEGJ=ARRAY(ISCTND,5)
      YSEGJ=ARRAY(ISCTND,6)
      IF(XSEGJ.EQ.XINSCT.AND.YSEGJ.EQ.YINSCT)GO TO 166
      IF(IRPT.EQ.2)GO TO 165
  164 ISCTND=IARRAY(2,ISEGJ,9)
      ISCTND=IABS(ISCTND)
      IRPT=2
      GO TO 163
  166 IFLAG=1
      XLEFT=XINSCT-.001*AAL
      XRIGHT=XINSCT+.001*AAL
      YLEFT=YINSCT-.001*BBL
      YRIGHT=YINSCT+.001*BBL
  167 IF(IRPT.EQ.1)IHIGHE=IARRAY(2,ISEGJ,9)
      IF(IRPT.EQ.2)IHIGHE=IARRAY(1,ISEGJ,9)
      IHIGHE=IABS(IHIGHE)
      XTEST=XINSCT+.001*(ARRAY(IHIGHE,5)-XINSCT)
      YTEST=YINSCT+.001*(ARRAY(IHIGHE,6)-YINSCT)
      TEMP1=XLEFT*YRIGHT-XRIGHT*YLEFT
      AL=TRISNS(XTEST,YTEST,XLEFT,YLEFT,XRIGHT,YRIGHT,TEMP1)
      IF(ABS(AL).LT.1.0E-8)GO TO 170
      IF(AL.LT.0.)GO TO 161
      GO TO 165
  170 CONTINUE
      ITSTNR=IWRKNR(1)
      CALL DELETE(IWRKNR,IFACE1)
      IF(ITSTNR.NE.IWRKNR(1))GO TO 171
      IWRKNR(1)=ITSTNR+2
      IWRKNR(ITSTNR+2)=IFACE1
      IWRKNR(ITSTNR+3)=IFACE2
      GO TO 161
  171 CONTINUE
      CALL DELETE(IWRKNR,IFACE2)
      GO TO 161
  306 XEND(JDSPLY)=XHIGH
      YEND(JDSPLY)=YHIGH
      CALL WINDOW
      GO TO 106
  201 CONTINUE
      IF(NTEMP.EQ.0)GO TO 103
      XNEAR=XHIGH
      YNEAR=YHIGH
      DO 203 J=1,3
  203 TEMPIN(J)=ARRAY(LHIGH,J)
      DPTHNR=ARRAY(LHIGH,4)
      CALL TSTNNP
  103 CONTINUE
      IF(IDRAW.EQ.1)GO TO 306
  106 IF(IARRAY(2,LHIGH,13).NE.0)GO TO 204
      IARRAY(1,LHIGH,13)=IWRKNR(1)
      IARRAY(2,LHIGH,13)=JPNTR+1
      IF(IWRKNR(1).EQ.0)GO TO 204
      IEND=IWRKNR(1)+1
      DO 202 J=2,IEND
      JPNTR=JPNTR+1
      LSTBFC(JPNTR)=IWRKNR(J)
  202 CONTINUE
  204 IARRAY(2,IL,9)=-LHIGH
  114 CONTINUE
  104 CONTINUE
      RETURN
C
      ENTRY DATSTR
      IEND=(N-1)/40
      DO 501 NPT=1,IEND
      ISTRT=(NPT-1)*40+1
      IFIN1=ISTRT+39
      IFIN=MIN0(IFIN1,N)
      WRITE(6,550)(I,(ARRAY(I,J),J=1,6),IARRAY(1,I,7),
     1(IARRAY(K,I,13),K=1,2),I=ISTRT,IFIN)
  501 CONTINUE
      IEND=(N-1)/64+1
      DO 503 NPT=1,IEND
      ISTRT=(NPT-1)*64+1
      IFIN1=ISTRT+63
      IFIN=MIN0(IFIN1,N)
      WRITE(6,555)(I,((IARRAY(J,I,K),J=1,2),K=8,12),I=ISTRT,
     1IFIN)
  503 CONTINUE
  550 FORMAT(1H1,8(/),15X,'COLUMN',5X,'1',10X,'2',10X,'3',
     110X,'4',10X,'5',10X,'6',8X,'7',6X,'13',/15X,'IDENT',
     23X,'THREE-DIMENSIONAL COORDINATES',5X,'DEPTH',5X,
     3'PICTURE PLANE COORD',1X,'CCV/CVX',1X,'NATURE',40(/15X
     4,I3,2X,6F11.4,3X,I2,4X,I2,I3))
  555 FORMAT(1H1,  ///15X,'COLUMN',4X,'8',10X,'9',9X,'10',
     19X,'11',9X,'12',/15X,'IDENT',2X,'FACES',6X,'LINE END',
     2' POINTS',5X,'INTERSECTING SURFACES',64(/15X,I3,2X,2I4
     3,3X,2I4,3X,2I4,3X,2I4,3X,2I4))
      WRITE(6,560)(LSTFAC(I),I=1,LSPNTR)
      WRITE(6,570)(LSTBFC(I),I=1,JPNTR)
      IEND=JDSPLY/64+1
      DO 502 J=1,IEND
      ISTRT=(J-1)*64+1
      IFIN1=ISTRT+63
      IFIN=MIN0(IFIN1,JDSPLY)
      WRITE(6,580)(XSTART(I),YSTART(I),XEND(I),YEND(I),
     1I=ISTRT,IFIN)
  502 CONTINUE
  560 FORMAT(////////41X,'LSTFAC',/5(/15X,5I5,5X,5I5))
  570 FORMAT(//41X,'LSTBFC',/5(/15X,5I5,5X,5I5))
  580 FORMAT(1H1,///,37X,'VISIBLE LINES',//21X,'XSTART',6X,
     2'YSTART',12X,'XEND',8X,'YEND',64(/15X,2F12.6,5X,
     32F12.6))
      RETURN
C
      ENTRY ALTER
      NPNTRA=NPNTR
      DO 471 II=1,LPNTR
      MIRROR=1
      LLOW=IARRAY(1,II,9)
      IF(LLOW.LT.0)GO TO 471
      IF(ARRAY(LLOW,4).GT.0.)MIRROR=MIRROR+1
      LHIGH=IARRAY(2,II,9)
      IF(ARRAY(LHIGH,4).GT.0.)MIRROR=MIRROR+2
      GO TO (471,481,482,472),MIRROR
  472 IARRAY(1,II,9)=-LLOW
      GO TO 471
  481 LPOINT=LLOW
      HPOINT=LHIGH
      GO TO 483
  482 LPOINT=LHIGH
      HPOINT=LLOW
  483 CONTINUE
      DO 485 I=1,3
      AA(I)=ARRAY(LPOINT,I)
  485 BB(I)=ARRAY(HPOINT,I)
      CALL PIERC(OBSPOS(1),OBSPOS(4),OBSPOS(7),AA,BB,PP)
      NPNTRA=NPNTRA+1
      DO 486 I=1,3
      ARRAY(NPNTRA,I)=PP(I)
  486 ARRAY(NPNTRA,I+3)=HMATRX(4,I)
      FCTR =HMATRX(4,4)
      DO 487 I=1,3
      IJ=I+3
      FCTR =FCTR+ARRAY(NPNTRA,I)*HMATRX(I,4)
      DO 487 J=1,3
  487 ARRAY(NPNTRA,IJ)=ARRAY(NPNTRA,IJ)+ARRAY(NPNTRA,J)*
     1HMATRX(J,I)
      DO 488 I=5,6
  488 ARRAY(NPNTRA,I)=ARRAY(NPNTRA,I)/FCTR
      MIRROR=MIRROR-1
      IARRAY(MIRROR,II,9)=NPNTRA
  471 CONTINUE
      IF(NPNTRA.NE.NPNTR)LRESTR=1
      RETURN
      END
      SUBROUTINE QSCRB1(N,ARRAY,IARRAY,FNORML,LSTFAC,LSTBFC,
     1IWRKNR,INTLVR,JGSET,XSTART,YSTART,XEND,YEND,/JDSPLY/)
      COMMON/SDFB/ALPHA,BETA,GAMMA,PSI,THETA,PHI,OBJMV(3),
     1VIEWMV(3),FOCUS,YOFF,ZOFF,SCLE,HMATRX(4,4),OBSPOS(12)
      COMMON /DSCRB/LPNTR,JPNTR,IFACE,TEMPIN(9),ICROSS,
     1INRSCT,ICOUNT,NTEMP,CURSEG,LLOW,LHIGH,XLOW,XHIGH,YLOW,
     2YHIGH,XNEAR,YNEAR,DPTHNR, AA(3),BB(3),CC(3),PP(3),AAL,
     3BBL,LRESTR
      INTEGER TPNTR,CURSEG,TEMP
      DIMENSION ARRAY(N,13),FNORML(N,3),LSTFAC(1),LSTBFC(1)
      INTEGER*2 IARRAY(2,N,13)
      DIMENSION IWRKNR(1),INTLVR(1),JGSET(1)
      DIMENSION XSTART(1),YSTART(1),XEND(1),YEND(1)
      DIMENSION AAA(3),BBB(3),AAAE(3),BBBE(3),TSTPNT(4)
      TRISNS(A0,B0,A1,B1,A2,B2,CONST)=A0*B1+A2*B0-A0*B2-
     1A1*B0+CONST
      ALPHA=0.
      BETA=0.
      GAMMA=0.0
      PSI=0.
      THETA=0.
      PHI=0.
      YOFF=0.
      ZOFF=0.
      FOCUS=1.5
      SCLE=0.5
      OBJMV(1)=-40.
      OBJMV(2)=0.
      OBJMV(3)=0.
      VIEWMV(1)=0.
      VIEWMV(2)=0.
      VIEWMV(3)=0.
      RETURN
C
      ENTRY VRTNTR
      MRPNTR=LPNTR+1
      ISTRT=MRPNTR+1
   86 NBR=0
      DO 92  IJ=1,LPNTR
      IFAC1=IARRAY(1,IJ,11)
      IFAC2=IARRAY(2,IJ,11)
      IFAC1=IABS(IFAC1)
      IF(IARRAY(1,IFAC1,8).LT.0.AND.IARRAY(1,IFAC2,8).LT.0.
     1OR.IFAC1.EQ.IFAC2)GO TO 92
      LLOWE=IARRAY(1,IJ,9)
      LLOWE=IABS(LLOWE)
      IF(ARRAY(LLOWE,4).GT.1.)GO TO 92
   98 LHIGHE=IARRAY(2,IJ,9)
      LHIGHE=IABS(LHIGHE)
      X1=ARRAY(LLOWE,5)
      X2=ARRAY(LHIGHE,5)
      XMAX=AMAX1(X1,X2)
      XMIN=AMIN1(X1,X2)
      IF(XMAX-XLOW)92,97,87
   87 IF(XMIN.GE.XLOW)GO TO 92
   88 Y1=ARRAY(LLOWE,6)
      Y2=ARRAY(LHIGHE,6)
      YMAX=AMAX1(Y1,Y2)
      IF(YMAX.LT.YLOW)GO TO 92
      TEMP1=X1*Y2-X2*Y1
      YMAX=YMAX+1.
      AL=TRISNS(XLOW,YLOW,X1,Y1,X2,Y2,TEMP1)
      AH=TRISNS(XLOW,YMAX,X1,Y1,X2,Y2,TEMP1)
      IF(AL.LT.0..AND.AH.LT.0..OR.AL.GT.0..AND.AH.GT.0.) GO
     1TO 92
C           MARK POLYGONS FOR INTERCEPT
      IEFAC1=IARRAY(1,IJ,11)
      IEFAC1=IABS(IEFAC1)
      IEFAC2=IARRAY(2,IJ,11)
      IF(IARRAY(1,IEFAC1,8).LT.0)GO TO 90
      DO 89  IK=ISTRT,MRPNTR
      IF(ISTRT.GT.MRPNTR)GO TO 89
      TEMP=IARRAY(1,IK,11)
      IF(TEMP.EQ.IEFAC1)GO TO 95
   89 CONTINUE
      MRPNTR=MRPNTR+1
      IARRAY(1,MRPNTR,11)=IEFAC1
      IARRAY(2,MRPNTR,11)=1
   90 IF(IARRAY(1,IEFAC2,8).LT.0) GO TO 92
      DO 91  IK=ISTRT,MRPNTR
      IF(ISTRT.GT.MRPNTR)GO TO 91
      TEMP=IARRAY(1,IK,11)
      IF(TEMP.EQ.IEFAC2)GO TO 96
   91 CONTINUE
      MRPNTR=MRPNTR+1
      IARRAY(1,MRPNTR,11)=IEFAC2
      IARRAY(2,MRPNTR,11)=1
   92 CONTINUE
      IARRAY(2,LLOW,13)=JPNTR+1
      DO 94  IJ=ISTRT,MRPNTR
      IF(ISTRT.GT.MRPNTR)GO TO 94
      IF(IARRAY(2,IJ,11)-IARRAY(2,IJ,11)/2*2.EQ.0)GO TO 94
      IFACE=IARRAY(1,IJ,11)
      DO 93  J=1,3
   93 TEMPIN(J)=ARRAY(LLOW,J)
      CALL PLNLIN(N,ARRAY,IARRAY,IFACE,LSTFAC,OBSPOS,TEMPIN,
     1PP)
      DVRTEX=ARRAY(LLOW,4)
      IF(ABS(DVRTEX).LT.0.10)GO TO 99
      PPX=DEPTH(PP(1),PP(2),PP(3))
      IF(ABS(PPX-DVRTEX).LT..01 )GO TO 94
      IF(PPX.LT.DVRTEX.OR.PPX.GT.0.005)GO TO 94
C          ENTER SET OF BLOCKING FACES
  101 JPNTR=JPNTR+1
      LSTBFC(JPNTR)=IARRAY(1,IJ,11)
      NBR=NBR+1
   94 CONTINUE
      IARRAY(1,LLOW,13)=NBR
      RETURN
   95 IARRAY(2,IK,11)=IARRAY(2,IK,11)+1
      GO TO 90
   96 IARRAY(2,IK,11)=IARRAY(2,IK,11)+1
      GO TO 92
   99 Y=ARRAY(LLOW,2)-OBSPOS(2)
      IF(ABS(Y).LT..1)GO TO 100
      IF(Y.GT.0..AND.PP(2).LT.0..OR.Y.LT.0..AND.PP(2).GT.0.)
     1GO TO 94
      IF(ABS(Y-PP(2)).LT..1)GO TO 94
      GO TO 101
  100 Z=ARRAY(LLOW,3)-OBSPOS(3)
      IF(ABS(Z).LT..1)GO TO 94
      IF(Z.GT.0..AND.PP(3).LT.0..OR.Z.LT.0..AND.PP(3).GT.0.)
     1GO TO 94
      IF(ABS(Z-PP(3)).LT..1)GO TO 94
      GO TO 101
   97 IF(XMAX.NE.XMIN)GO TO 88
      ARRAY(LLOWE,5)=ARRAY(LLOWE,5)-.00001
      ARRAY(LHIGHE,5)=ARRAY(LHIGHE,5)+.00001
      MRPNTR=LPNTR+1
      GO TO 86
C
      ENTRY TSTNNP
C          TEST POINT NEAR TO KNOWN POINT
C
C          G IS THE SET OF ON VIEW FACES OF WHICH THE NODE
C          BEING TESTED IS A MEMBER
C          J IS THE SET OF ON VIEW FACES BELONGING TO THE
C          INTERSECTING SEGMENTS
C          FIND SET OF FACES WHICH MAKE UP J AND G
      JGSET(1)=0
  115 ISEGJ=INTLVR(NTEMP)
      IFACE1=IARRAY(1,ISEGJ,11)
      IFACE1=IABS(IFACE1)
      IFACE2=IARRAY(2,ISEGJ,11)
      IF(IARRAY(1,IFACE1,8).LT.0)GO TO 113
      CALL MERGE(JGSET,IFACE1)
  113 IF(IARRAY(1,IFACE2,8).LT.0) GO TO 114
      CALL MERGE(JGSET,IFACE2)
  114 NTEMP=NTEMP-1
      IF(NTEMP.GT.0)GO TO 115
      IEND=JGSET(1)+1
      DO 117 IT=2,IEND
      IF(IEND.LE.1)GO TO 117
      IFACE=JGSET(IT)
      NSEGS=IARRAY(1,IFACE,8)
      TPNTR=IARRAY(2,IFACE,8)
C           TEST NEAR POINT WITH RESPECT TO SETS J AND G
C           SEARCH FACE IFACE FOR INTERSECTION
  121 LLOWE=LSTFAC(TPNTR+NSEGS-1)
      LHIGHE=LSTFAC(TPNTR)
      ICROSS=0
      DO 122 IK=1,NSEGS
      X1=ARRAY(LLOWE,5)
      X2=ARRAY(LHIGHE,5)
      XMAX=AMAX1(X1,X2)
      XMIN=AMIN1(X1,X2)
      IF(XMAX-XNEAR)122,123,124
  124 IF(XMIN.GE.XNEAR)GO TO 122
  126 Y1=ARRAY(LLOWE,6)
      Y2=ARRAY(LHIGHE,6)
      YMAX=AMAX1(Y1,Y2)
      IF(YMAX.LT.YNEAR)GO TO 122
      TEMP1=X1*Y2-X2*Y1
      YMAX=YMAX+1
      AL=TRISNS(XNEAR,YNEAR,X1,Y1,X2,Y2,TEMP1)
      AH=TRISNS(XNEAR,YMAX,X1,Y1,X2,Y2,TEMP1)
      IF(AL.LT.0..AND.AH.GT.0..OR.AL.GT.0..AND.AH.LT.0.)
     1ICROSS=ICROSS+1
      LLOWE=LHIGHE
      LHIGHE=LSTFAC(TPNTR+IK)
  122 CONTINUE
      GO TO 116
  123 IF(XMAX.NE.XMIN)GO TO 126
      ARRAY(LLOWE,5)=ARRAY(LLOWE,5)-.00001
      ARRAY(LHIGHE,5)=ARRAY(LHIGHE,5)+.00001
      GO TO 121
  116 IF(ICROSS-ICROSS/2*2.NE.0)GO TO 118
      CALL DELETE(IWRKNR,IFACE)
      GO TO 120
  118 CONTINUE
C           POINT LIES INSIDE POLYGON-SEARCH SET
      IEND=IWRKNR(1)+1
      DO 119 IK=2,IEND
      IF(IEND.LE.1)GO TO 119
      IF(IFACE.EQ.IWRKNR(IK))GO TO 117
  119 CONTINUE
      CALL PLNLIN(N,ARRAY,IARRAY,IFACE,LSTFAC,OBSPOS,TEMPIN,
     1PP)
      DPTHPP=DEPTH(PP(1),PP(2),PP(3))
      IF(DPTHPP.LT.DPTHNR) GO TO 117
      IWRKNR(IWRKNR(1)+2)=IFACE
      IWRKNR(1)=IWRKNR(1)+1
  120 ICOUNT=ICOUNT-1
  117 CONTINUE
      RETURN
C
      ENTRY LNTRSC
C           THIS SUBROUTINE CALCULATES THE INTERSECTION OF ALL
C           SEGMENTS WITH A GIVEN SEGMENT IN THE PICTURE PLANE
      IXPNTR=LPNTR+1
      JGSET(1)=0
      INRSCT=0
      IOVRLP=0
      IFACE1=IARRAY(1,CURSEG,11)
      IFACE1=IABS(IFACE1)
      IFACE2=IARRAY(2,CURSEG,11)
      IF(ABS(AAL).LT..0001)GO TO 68
      CRNTSL=BBL/AAL
   61 TEMP2=YLOW-CRNTSL*XLOW
      TEMP3=XLOW*YHIGH-XHIGH*YLOW
      X1=ARRAY(LLOW,1)
      Y1=ARRAY(LLOW,2)
      Z1=ARRAY(LLOW,3)
      X2=ARRAY(LHIGH,1)
      Y2=ARRAY(LHIGH,2)
      Z2=ARRAY(LHIGH,3)
      TEMP1=X1*Y2-X2*Y1
      TEMP5=Y1*Z2-Y2*Z1
      TEMP6=Z1*X2-Z2*X1
      DO 702 I=1,3
      AAA(I)=ARRAY(LLOW,I)
  702 BBB(I)=ARRAY(LHIGH,I)
      IEND=LPNTR
      DO 67 IEXAM=1,IEND
      IFAC1=IARRAY(1,IEXAM,11)
      IFAC1=IABS(IFAC1)
      IFAC2=IARRAY(2,IEXAM,11)
      IF(IFAC1.EQ.IFACE1.OR.IFAC2.EQ.IFACE1.OR.IFAC1.EQ.
     1IFACE2.OR.IFAC2.EQ.IFACE2.OR.IFAC1.EQ.IFAC2.OR.
     2IARRAY(1,IFAC1,8).LT.0.AND.IARRAY(1,IFAC2,8).LT.0)
     1GO TO 67
      LLOWE=IARRAY(1,IEXAM,9)
      LLOWE=IABS(LLOWE)
      IF(ARRAY(LLOWE,4).GT.1.)GO TO 67
      LHIGHE=IARRAY(2,IEXAM,9)
      LHIGHE=IABS(LHIGHE)
      XLOWE=ARRAY(LLOWE,5)
      YLOWE=ARRAY(LLOWE,6)
      XHIGHE=ARRAY(LHIGHE,5)
      YHIGHE=ARRAY(LHIGHE,6)
C         TEST FOR INTERSECTION USING TRI-SENSE TEST
      AL=TRISNS(XLOWE,YLOWE,XLOW,YLOW,XHIGH,YHIGH,TEMP3)
      AH=TRISNS(XHIGHE,YHIGHE,XLOW,YLOW,XHIGH,YHIGH,TEMP3)
C          ARE POINTS COLLINEAR
      INDAB1=1
      IF(ABS(AL).LT.1.E-5)INDAB1=INDAB1+1
      IF(ABS(AH).LT.1.E-5)INDAB1=INDAB1+2
      GO TO (611,613,614,72),INDAB1
  613 ICOMN=LLOWE
      IDIFF=LHIGHE
      GO TO 612
  614 ICOMN=LHIGHE
      IDIFF=LLOWE
      GO TO 612
  611 IF(AL.LT.0..AND.AH.LT.0..OR.AL.GT.0..AND.AH.GT.0.)
     1GO TO 67
  612 TEMP4=XLOWE*YHIGHE-XHIGHE*YLOWE
      AL=TRISNS(XLOW,YLOW,XLOWE,YLOWE,XHIGHE,YHIGHE,TEMP4)
      AH=TRISNS(XHIGH,YHIGH,XLOWE,YLOWE,XHIGHE,YHIGHE,TEMP4)
      IF(INDAB1.NE.1)GO TO 615
      INDAB2=1
      IF(ABS(AL).LT.1.E-5)INDAB2=INDAB2+1
      IF(ABS(AH).LT.1.E-5)INDAB2=INDAB2+2
      GO TO (615,616,617),INDAB2
  616 HINSCT=XLOW
      VINSCT=YLOW
      GO TO 701
  617 HINSCT=XHIGH
      VINSCT=YHIGH
      GO TO 701
  615 IF(AL.LT.0..AND.AH.LT.0..OR.AL.GT.0..AND.AH.GT.0.)
     1GO TO 67
      IF(INDAB1.NE.1)GO TO 620
  701 CONTINUE
```

```
      DO 703 I=1,3
      AAAE(I)=ARRAY(LLOWE,I)
  703 BBBE(I)=ARRAY(LHIGHE,I)
      CALL PIERC(AAA,OBSPOS(10),BBB,AAAE,BBBE,PP)
      IXX=1
  704 GO TO (711,712,713,67),IXX
  711 AL=TRISNS(OBSPOS(10),OBSPOS(11),X1,Y1,X2,Y2,TEMP1)
      AH=TRISNS(PP(1),PP(2),X1,Y1,X2,Y2,TEMP1)
      GO TO 714
  712 AL=TRISNS(OBSPOS(11),OBSPOS(12),Y1,Z1,Y2,Z2,TEMP5)
      AH=TRISNS(PP(2),PP(3),Y1,Z1,Y2,Z2,TEMP5)
      GO TO 714
  713 AL=TRISNS(OBSPOS(12),OBSPOS(10),Z1,X1,Z2,X2,TEMP6)
      AH=TRISNS(PP(3),PP(1),Z1,X1,Z2,X2,TEMP6)
  714 IXX=IXX+1
      IF(ABS(AL).LT.FOCUS)GO TO 704
      IF(AL.LT.0..AND.AH.GT.0..OR.AL.GT.0..AND.AH.LT.0.) GO
     1TO 67
  626 AALE=XHIGHE-XLOWE
      BBLE=YHIGHE-YLOWE
      IF(ABS(AALE).LT..0001)GO TO 69
      EXAMSL=BBLE/AALE
      IF(INDAB2.NE.1)GO TO 64
   62 HINSCT=(TEMP2+EXAMSL*XLOWE-YLOWE)/(EXAMSL-CRNTSL)
   63 VINSCT=TEMP2+CRNTSL*HINSCT
   64 CONTINUE
      IXPNTR=IXPNTR+1
      INRSCT=INRSCT+1
      FNORML(INRSCT,1)=HINSCT
      FNORML(INRSCT,2)=VINSCT
      IARRAY(1,IXPNTR,9)=IEXAM
      IF(IOVRLP) 81,67,76
  620 IF(JGSET(1).EQ.0)GO TO 622
      JEND=JGSET(1)+1
      DO 621 ISR=2,JEND
      IF(ICOMN.EQ.JGSET(ISR))GO TO 625
  621 CONTINUE
  622 JGSET(1)=JGSET(1)+1
      JGSET(JGSET(1)+1)=ICOMN
      INTLVR(JGSET(1))=IDIFF
      GO TO 67
  625 XO=ARRAY(IDIFF,5)
      YO=ARRAY(IDIFF,6)
      MARK=INTLVR(ISR-1)
      XT=ARRAY(MARK,5)
      YT=ARRAY(MARK,6)
      AL=TRISNS(XO,YO,XLOW,YLOW,XHIGH,YHIGH,TEMP3)
      AH=TRISNS(XT,YT,XLOW,YLOW,XHIGH,YHIGH,TEMP3)
      IF(AL.LT.0..AND.AH.LT.0..OR.AL.GT.0..AND.AH.GT.0.)GO
     1TO 621
      GO TO 701
   68 CRNTSL=10000.
      GO TO 61
   69 EXAMSL=10000.
      GO TO 62
C           OVERLAP HAS BEEN ESTABLISHED
C           CHECK FOR END POINTS OF COMMON SEGMENT
   72 IF(CRNTSL.GT.1000.)GO TO 77
      RIGHT=AMIN1(AMAX1(XLOW,XHIGH),AMAX1(XLOWE,XHIGHE))
      ALEFT=AMAX1(AMIN1(XLOW,XHIGH),AMIN1(XLOWE,XHIGHE))
      IF(RIGHT-ALEFT)67,74,75
   75 IOVRLP=1
   74 HINSCT=RIGHT
      GO TO 63
   76 HINSCT=ALEFT
      IOVRLP=0
      GO TO 63
C           SLOPE IS VERY HIGH USE VERTICAL COORDINATES
   77 RIGHT=AMIN1(AMAX1(YLOW,YHIGH),AMAX1(YLOWE,YHIGHE))
      ALEFT=AMAX1(AMIN1(YLOW,YHIGH),AMIN1(YLOWE,YHIGHE))
      IF(RIGHT-ALEFT)67,79,80
   80 IOVRLP=-1
   79 VINSCT=RIGHT
      HINSCT=XLOW
      GO TO 64
   81 VINSCT=ALEFT
      IOVRLP=0
      GO TO 64
   67 CONTINUE
      RETURN
C
      ENTRY WINDOW
      IX1=XSTART(JDSPLY)
      IX2=XEND(JDSPLY)
      IY1=YSTART(JDSPLY)
      IY2=YEND(JDSPLY)
      IF(IX1.GE.1.AND.IX2.GE.1)GO TO 461
      IF(IX1.LE.-1.AND.IX2.LE.-1)GO TO 461
      IF(IY1.GE.1.AND.IY2.GE.1)GO TO 461
      IF(IY1.LE.-1.AND.IY2.LE.-1)GO TO 461
      IAGN=1
      X=XSTART(JDSPLY)
      Y=YSTART(JDSPLY)
      IX=IX1
      IY=IY1    -  0  +
  401 IF(IX)411,412,413
  411 VINSCT=Y+CRNTSL*(-1.-X)
  421 IF(ABS(VINSCT).LE.1.)GO TO 426
  423 IF(IY)424,461,425
  426 XTEMP=-1.
      GO TO 447
  424 HINSCT=X+(-1.-Y)/CRNTSL
      YTEMP=-1.
  422 IF(ABS(HINSCT).GT.1.)GO TO 461
      XTEMP=HINSCT
      GO TO 450
  425 HINSCT=X+(1.-Y)/CRNTSL
      YTEMP=1.
      GO TO 422
  412 IF(IY)424,432,425
  432 XTEMP=X
      YTEMP=Y
      GO TO 450
  413 VINSCT=Y+CRNTSL*(1.-X)
      IF(ABS(VINSCT).LE.1.)GO TO 446
      GO TO 423
  446 XTEMP=1.
  447 YTEMP=VINSCT
  450 GO TO (451,460),IAGN
  451 X=XEND(JDSPLY)
      Y=YEND(JDSPLY)
      IX=IX2
      IY=IY2
      IAGN=2
      XSTART(JDSPLY)=XTEMP
      YSTART(JDSPLY)=YTEMP
      GO TO 401
  460 CONTINUE
      XEND(JDSPLY)=XTEMP
      YEND(JDSPLY)=YTEMP
      RETURN
  461 JDSPLY=JDSPLY-1
      RETURN
      END


      SUBROUTINE PLNLIN(N,ARRAY,IARRAY,IFACE,LSTFAC,OBSPOS,
     1TEMPIN,PP)
      INTEGER*2 IARRAY
      DIMENSION ARRAY(N,13),IARRAY(2,N,13),LSTFAC(1),
     1OBSPOS(1),TEMPIN(1),AA(3),BB(3),CC(3),PP(1)
      NTTL=IARRAY(1,IFACE,8)
      LOC=IARRAY(2,IFACE,8)
      IPOINT=LSTFAC(LOC+NTTL-1)
      DO 98  J=1,3
   98 AA(J)=ARRAY(IPOINT,J)
      IPOINT=LSTFAC(LOC)
      DO 99  J=1,3
   99 BB(J)=ARRAY(IPOINT,J)
      IPOINT=LSTFAC(LOC+1)
      DO 100 J=1,3
  100 CC(J)=ARRAY(IPOINT,J)
      CALL PIERC(AA,BB,CC,OBSPOS,TEMPIN,PP)
      RETURN
      END


      SUBROUTINE PIERC(P1,P2,P3,PL1,PL2,PP)
      DIMENSION P1(3),P2(3),P3(3),PL1(3),PL2(3),PP(3)
C         THIS SUBROUTINE CALCULATES THE INTERSECTION OF A
C           PLANE DETERMINED BY POINTS P1,P2,P3 AND A LINE
C           WHICH CONTAINS POINTS PL1 AND PL2.
C           THE COORDINATES OF THE INTERSECTION ARE RETURNED
C           IN THE ARRAY PP
C           THE DIRECTION NUMBERS OF THE NORMAL TO THE PLANE
C           ARE A, B, C.
      A=(P2(2)-P1(2))*(P3(3)-P1(3))-(P2(3)-P1(3))*(P3(2)-
     1P1(2))
      B=(P2(3)-P1(3))*(P3(1)-P1(1))-(P2(1)-P1(1))*(P3(3)-
     1P1(3))
      C=(P2(1)-P1(1))*(P3(2)-P1(2))-(P2(2)-P1(2))*(P3(1)-
     1P1(1))
      D=-A*P1(1)-B*P1(2)-C*P1(3)
      AL=PL2(1)-PL1(1)
      BL=PL2(2)-PL1(2)
      CL=PL2(3)-PL1(3)
      T=-(A*PL1(1)+B*PL1(2)+C*PL1(3)+D)/(A*AL+B*BL+C*CL)
      PP(1)=AL*T+PL1(1)
      PP(2)=BL*T+PL1(2)
      PP(3)=CL*T+PL1(3)
      RETURN
      END


      FUNCTION DEPTH(X,Y,Z)
      COMMON/SDFB/ALPHA,BETA,GAMMA,PSI,THETA,PHI,OBJMV(3),
     1VIEWMV(3),FOCUS,YOFF,ZOFF,SCLE,HMATRX(4,4),OBSPOS(12)
      DEPTH=X*HMATRX(1,1)+Y*HMATRX(2,1)+Z*HMATRX(3,1)+
     1HMATRX(4,1)
      RETURN
      END


      SUBROUTINE DELETE(LIST,ITEM)
      DIMENSION LIST(1)
      N=LIST(1)
      IEND=N+1
      DO 1 I=2,IEND
      IF(IEND.EQ.1)GO TO 1
      IF(ITEM.EQ.LIST(I))GO TO 2
    1 CONTINUE
      RETURN
    2 IF(I.EQ.IEND)GO TO 3
      LIST(I)=LIST(IEND)
    3 LIST(1)=N-1
      RETURN
      END


      SUBROUTINE MERGE(LIST,ITEM)
      DIMENSION LIST(1)
      N=LIST(1)
      IF(N.EQ.0)GO TO 2
      IEND=N+1
      DO 1 I=2,IEND
      IF(ITEM.EQ.LIST(I))GO TO 2
    1 CONTINUE
      LIST(N+2)=ITEM
      LIST(1)=N+1
    2 CONTINUE
      RETURN
      END


      SUBROUTINE READCR(N,TEMPIN)
      DIMENSION TEMPIN(3)
      IF(N.EQ.0)GO TO 1
      READ(5,2)N
      RETURN
    1 READ(5,3)(TEMPIN(I),I=1,3)
      RETURN
    2 FORMAT(1I10)
    3 FORMAT(3F20.6)
      END


      SUBROUTINE CROSS(A,B,C)
      DIMENSION A(3),B(3),C(3)
C         THIS SUBROUTINE CALCULATES THE CROSS PRODUCT
C         A X B FOR 3 DIMENSIONAL VECTORS
      C(1)=A(2)*B(3)-A(3)*B(2)
      C(2)=A(3)*B(1)-A(1)*B(3)
      C(3)=A(1)*B(2)-A(2)*B(1)
      RETURN
      END


      SUBROUTINE LABEL
      COMMON/SDFB/ALPHA,BETA,GAMMA,PSI,THETA,PHI,OBJMV(3),
     1VIEWMV(3),FOCUS,YOFF,ZOFF,SCLE,HMATRX(4,4),OBSPOS(12)
      WRITE(6,1)ALPHA,BETA,GAMMA,PSI,THETA,PHI,(OBJMV(I),
     1I=1,3),(VIEWMV(I),I=1,3),YOFF,ZOFF,FOCUS,SCLE
    1 FORMAT(4X,' ALPHA=',1F12.6,8X,'   BETA=',1F12.6,9X,
     1'GAMMA=',1F12.6,/7X,'PSI=',1F12.6,8X,' THETA=',1F12.6,
     28X,'    PHI=',1F12.6,/2X,'OBJMV(1)=',1F12.6,6X,
     3'OBJMV(2)=',1F12.6,6X,'OBJMV(3)=',1F12.6,/1X,
     4'VIEWMV(1)=',1F12.6,5X,'VIEWMV(2)=',1F12.6,5X,
     5'VIEWMV(3)=',1F12.6,/3X,'YOFF=',1F12.6,5X,'   ZOFF=',
     61F12.6,5X,'FOCUS=',1F12.6,5X,'SCALE=',1F12.6)
      WRITE(6,595)OBJMV(1),OBJMV(2),OBJMV(3),ALPHA,BETA,
     1GAMMA,VIEWMV(1),VIEWMV(2),VIEWMV(3)
      WRITE(6,596)PSI,THETA,PHI,FOCUS,SCLE,YOFF,ZOFF
  595 FORMAT(1H1,13(/),18X,'PARAMETERS FOR THIS VIEW',
     1///15X,'OBJECT POSITION (1)',15X,F10.1,
     2//15X,'OBJECT POSITION (2)',15X,F10.1,
     3//15X,'OBJECT POSITION (3)',15X,F10.1,
     4// 15X,'OBJECT ANGLE ALPHA',16X,F10.1,
     5// 15X,'OBJECT ANGLE BETA', 17X,F10.1,
     6// 15X,'OBJECT ANGLE GAMMA',16X,F10.1,
     7// 15X,'OBSERVERS POSITION (1)',12X,F10.1,
     8// 15X,'OBSERVERS POSITION (2)',12X,F10.1,
     9// 15X,'OBSERVERS POSITION (3)',12X,F10.1)
  596 FORMAT(/15X,'OBSERVERS ANGLE PSI',15X,F10.1,
     1// 15X,'OBSERVERS ANGLE THETA',13X,F10.1,
     2// 15X,'OBSERVERS ANGLE PHI',15X,F10.1,
     3// 15X,'FOCAL LENGTH',22X,F10.1,
     4// 15X,'SCALE',29X,F10.1,
     5// 15X,'HORIZONTAL OFFSET',17X,F10.1,
     6// 15X,'VERTICAL OFFSET',19X,F10.1)
      RETURN
      END
      SUBROUTINE DRAWIT(NLINES,XSTART,YSTART,XEND,YEND,IR)
      REAL*4 TITLE1(3)/12HCARRIER (1) /
      REAL*4 TITLE2(3)/12HCARRIER (2) /
      REAL*4 TITLE3(3)/12HCARRIER (3) /
      REAL*4 TITLE4(3)/12HCARRIER (4) /
      REAL*4 TITLE5(3)/12HCARRIER (5) /
      REAL XWNDOW(5)/3.,-3.,-3.,3.,3./
      REAL YWNDOW(5)/3.,3.,-3.,-3.,3./
      REAL*4 IDENT(5)/20HR. B. DESENS - BOX D/
      DIMENSION XSTART(1),YSTART(1),XEND(1),YEND(1)
      CALL PLOTS
      CALL PLOT(4.,6.,-3)
      CALL PLOT(XWNDOW(1),YWNDOW(1),3)
      DO 2 I=2,5
      CALL PLOT(XWNDOW(I),YWNDOW(I),2)
    2 CONTINUE
      DO 1 I=1,NLINES
      XSTART(I)=XSTART(I)*3.
      YSTART(I)=YSTART(I)*3.
      XEND(I)=XEND(I)*3.
      YEND(I)=YEND(I)*3.
      CALL PLOT(XSTART(I),YSTART(I),3)
      CALL PLOT(XEND(I),YEND(I),2)
    1 CONTINUE
      GO TO (15,14,13,12,11),IR
   21 CONTINUE
      CALL SYMBOL(-3.0,5.0,0.14,IDENT,0.0,20)
      CALL PLOT(-4.0,8.0,-3)
      CALL PLOTE
      RETURN
   11 CALL SYMBOL(-1.2,-3.4,0.28,TITLE1,0.0,12)
      GO TO 21
   12 CALL SYMBOL(-1.2,-3.4,0.28,TITLE2,0.0,12)
      GO TO 21
   13 CALL SYMBOL(-1.2,-3.4,0.28,TITLE3,0.0,12)
      GO TO 21
   14 CALL SYMBOL(-1.2,-3.4,0.28,TITLE4,0.0,12)
      GO TO 21
   15 CALL SYMBOL(-1.2,-3.4,0.28,TITLE5,0.0,12)
      GO TO 21
      END


      SUBROUTINE DRAWVP(NL,X1,Y1,X2,Y2)
      DIMENSION X1(1),Y1(1),X2(1),Y2(1)
      DIMENSION DRAW(10,200),JXY(200)
C        COLUMN DIMENSION OF DRAW AND DIMENSION OF JXY MUST
C        BE GREATER THAN OR EQUAL TO  TWO TIMES THE NUMBER
C        OF LIMES TO BE DRAWN
C        ISTEP IS THE NUMBER OF SYMBOLS PER LINE
C        ND IS THE ROW DIMENSION OF ARRAY DRAW
      ISTEP=10
      ND=10
      DO 1 J=1,NL
      JJ=J*2
      DRAW(1,JJ-1)=X1(J)
      DRAW(1,JJ  )=Y1(J)
      XSTEP=(X2(J)-X1(J))/ISTEP
      YSTEP=(Y2(J)-Y1(J))/ISTEP
      DO 1 I=2,ISTEP
      DRAW(I,JJ-1)=DRAW(I-1,JJ-1)+XSTEP
      DRAW(I,JJ  )=DRAW(I-1,JJ  )+YSTEP
    1 CONTINUE
      NLTT=NL*2
      DO 2 I=1,NLTT
    2 JXY(I)=I
      CALL VPLOT(DRAW,JXY,ISTEP,ND,NL,1,-1.,1.,-1.,1.)
      RETURN
      END

      SUBROUTINE VPLOT(XY,JXY,N,NDIM,NCUR,ISCALE,XL,XU,YL,
     1YU)
      DIMENSION IGRID(101),XS(11),YS(13),ICHAR(7),XY(1)
      DIMENSION JXY(1)
      DATA ICHAR/1H+,1H*,1H.,1H$,1H=,1H.,1H /
      XS(1)=XL
      XMAX=XU
      YMIN=YL
      YS(1)=YU
      IF(ISCALE.NE.0)GO TO 32
      XMAX=-1.0E 20
      XS(1)=-XMAX
      YS(1)=XMAX
      YMIN=XS(1)
      J2=0
      DO 31 J=1,NCUR
      J2=J2+2
      JIX=(JXY(J2-1)-1)*NDIM
      JIY=(JXY(J2)-1)*NDIM
      DO 31 I=1,N
      IJX=JIX+I
      IJY=JIY+I
      IF(XY(IJX).GT.XMAX)XMAX=XY(IJX)
      IF(XY(IJX).LT.XS(1))XS(1)=XY(IJX)
      IF(XY(IJY).GT.YS(1))YS(1)=XY(IJY)
      IF(XY(IJY).LT.YMIN)YMIN=XY(IJY)
   31 CONTINUE
   32 XR=XMAX-XS(1)
      IF(XR.EQ.0.0)XR=1.0E-20
      YR=YS(1)-YMIN
      IF(YR.EQ.0.0)YR=1.0E-20
      XT=XMAX*XS(1)
      YT=YMIN*YS(1)
      IF(XT.LT.0.0)IYAX=100.0*(-XS(1))/XR+1.5
      IF(YT.LE.0.0)IXAX=48.0*YS(1)/YR+1.5
      XMAX=XR/10.
      DO 46 I=2,11
   46 XS(I)=XS(I-1)+XMAX
      XMAX=YR/12.
      DO 47 I=2,13
   47 YS(I)=YS(I-1)-XMAX
      PRINT 10,  (XS(I),I=1,11)
      II=1
      KK=0
      DO 146 LINE=1,49
      DO 101 J=1,101
  101 IGRID(J)=ICHAR(7)
      IF(YT.GT.0.0)GO TO 109
      IF(LINE.NE.IXAX)GO TO 109
      DO 105 J=1,101
  105 IGRID(J)=ICHAR(6)
  109 IF(XT.LT.0.0)IGRID(IYAX)=ICHAR(6)
      J2=0
      DO 125 J=1,NCUR
      J2=J2+2
      JIX=(JXY(J2-1)-1)*NDIM
      JIY=(JXY(J2)-1)*NDIM
      JC=MOD(J,5)
      DO 125 I=1,N
      IJX=JIX+I
      IJY=JIY+I
      IPTY=48.0*(YS(1)-XY(IJY))/YR+1.5
      IF(IPTY.GT.49)IPTY=49
      IF(IPTY.LT.1)IPTY=1
      IF(IPTY.NE.LINE)GO TO 125
      IPTX=100.0*(XY(IJX)-XS(1))/XR+1.5
      IF(IPTX.LT.1)IPTX=1
      IF(IPTX.GT.101)IPTX=101
      IF(JC.NE.0)GO TO 119
      IGRID(IPTX)=ICHAR(5)
      GO TO 125
  119 IGRID(IPTX)=ICHAR(JC)
  125 CONTINUE
      IF(KK.GT.0)GO TO 134
      PRINT 20,   YS(II),(IGRID(I),I=1,101),YS(II)
      II=II+1
      GO TO 135
  134 PRINT 30,   (IGRID(I),I=1,101)
  135 KK=KK+1
      IF(KK.NE.4)GO TO 146
      KK=0
  146 CONTINUE
      PRINT 40,(XS(I),I=1,11)
      RETURN
   10 FORMAT(1H1,1PE15.2,10E10.2/10X,1H*,20(5H+****),2H+*)
   20 FORMAT(1PE10.2,1H+,101A1,1H+,E9.2)
   30 FORMAT(10X,1H*,101A1,1H*)
   40 FORMAT(10X,1H*,20(5H+****),2H+*/1PE16.2,10E10.2)
      END
```

REPRESENTATIVE DATA

FIRST TWO FACES OF AIRCRAFT CARRIER

```
          36
   0.0                  45.6                57.0
   0.0                   0.0                57.0
   0.0                 -45.6                57.0
 353.4                 -64.0                57.0
 365.0                -131.0                57.0
 955.0                -131.0                57.0
1009.0                 -79.8                57.0
1068.0                 -68.5                57.0
1046.0                  62.7                57.0
1052.0                  45.6                57.0
 353.0                 -64.0                57.0
1052.0                  45.6                57.0
1058.5                   5.7                57.0
 358.5                -103.9                57.0
1058.5                   5.7                57.0
1065.0                 -33.5                57.0
 450.5                -131.0                57.0
1065.0                 -33.5                57.0
1068.0                 -68.5                57.0
1065.0                 -33.5                57.0
1058.5                   5.7                57.0
1052.0                  45.6                57.0
1046.0                  62.7                57.0
 930.0                  50.2                57.0
 930.0                  85.6                57.0
 828.0                  90.0                57.0
 828.0                 114.0                57.0
 647.0                 114.0                57.0
 647.0                 110.0                57.0
 694.0                 110.0                57.0
 694.0                  52.5                57.0
 647.0                  52.5                57.0
 647.0                 110.0                57.0
 647.0                 114.0                57.0
 394.0                 114.0                57.0
 302.0                  63.0                57.0
           4
 694.0                  52.5                57.0
 694.0                 110.0                57.0
 694.0                 110.0                79.8
 694.0                  52.5                79.8
```

COMPUTER PROGRAM FOR THE SDS-9300

```
      COMMON/SDFB/ALPHA,BETA,GAMMA,PSI,THETA,PHI,OBJMV(3),
     1VIEWMV(3),FOCUS,YOFF,ZOFF,SCLE,HMATRX(4,4),OBSPOS(12)
      DIMENSION ARRAY(80,13),IARRAY(2,80,13),FNORML(80,3)
      DIMENSION LSTFAC(250),LSTBFC(1)
      DIMENSION XSTART(80),YSTART(80),XEND(80),YEND(80)
      EQUIVALENCE (ARRAY(1,1),IARRAY(1,1,1))
      EQUIVALENCE (FNORML(1,3),LSTBFC(1))
      EXTERNAL READCR
      N=80
      NFACES=32
      CALL DESCRB(ARRAY,IARRAY,FNORML,LSTFAC,N,READCR,
     1XSTART,YSTART,XEND,YEND,NLINES,LSTBFC,NUSED)
      DO 1 J=1,NFACES
      CALL FACES
    1 CONTINUE
      CALL LINES
      CALL CCVCVX
    2 CONTINUE
      CALL CONV3D(N,ARRAY,NUSED)
      CALL HDNSRF
      CALL ALTER
      CALL CMPUT
      CALL DATSTR
      CALL DRAWVP(NLINES,XSTART,YSTART,XEND,YEND)
      CALL LABEL
      IF(SENSE SWITCH 4)9,3
    3 CONTINUE
      PAUSE
      CALL ERASE
      READ(5,5)ALPHA,BETA,GAMMA,(OBJMV(I),I=1,3)
      READ(5,5)PSI,THETA,PHI,(VIEWMV(J),J=1,3)
      READ(5,10)FOCUS,SCALE,YOFF,ZOFF
      GO TO 2
    5 FORMAT(6F10.3)
   10 FORMAT(4F10.6)
    9 CONTINUE
      END


      SUBROUTINE CONV3D(N,PNTLST,NUSED)
      COMMON/SDFB/ALPHA,BETA,GAMMA,PSI,THETA,PHI,OBJMV(3),
     1VIEWMV(3),FOCUS,YOFF,ZOFF,SCLE,HMATRX(4,4),OBSPOS(12)
      DIMENSION PNTLST(N,13),DISPLY(3),HCRD3D(4),TEMP(4)
C           THIS PROGRAM USES THE FOURTH COORDINATE OF THE
C           HOMOGENEOUS SET EQUAL TO 1.0 AND THEREFORE REDUCES
C           THE NUMBER OF MULTIPLICATIONS NECESSARY
      CALL VMVMAT
      TEMP(4)=1.0
      DO 5 I=1,NUSED
      DO 1 IN=1,4
    1 HCRD3D(IN)=0.0
C           CONVERT THREE-DIMENSIONAL COORDINATES TO FOUR-
C           DIMENSIONAL HOMOGENEOUS COORDINATES
      DO 2 IM=1,3
    2 TEMP(IM)=PNTLST(I,IM)
      DO 3 K=1,4
      DO 3 J=1,4
      HCRD3D(K)=HCRD3D(K)+TEMP(J)*HMATRX(J,K)
    3 CONTINUE
      DO 4 ID=2,3
      DISPLY(ID)=HCRD3D(ID)/HCRD3D(4)
    4 CONTINUE
C     SAVE DEPTH INFORMATION
      DISPLY(1)=HCRD3D(1)
      DO 5 IS=1,3
    5 PNTLST(I,IS+3)=DISPLY(IS)
      END
      RETURN
      SUBROUTINE VMVMAT
      COMMON/SDEB/ALPHA,BETA,GAMMA,PSI,THETA,PHI,OBJMV(3),
     1VIEWMV(3),FOCUS,YOFF,ZOFF,SCLE,HMATRX(4,4),OBSPOS(12)
      DIMENSION ROTMAT(4,4),ORIMAT(4,4),TEMP(3)
      DIMENSION TEMPY(3),TEMPZ(3),TEMPF(3)
C         THIS SUBROUTINE CALCULATES THE 4 X 4 MATRIX FOR
C         PERSPCTIVE TRANSFORMATIONS
C
C                                              .W
C                               .Z             .
C                               .              .
C                               .        . . . .
C                               .        U      .
C                               .        OBJECT  .
C                   . . . . . . .        AXES     .V
C                   X             .
C                      VIEWPLANE    .
C                        AXES         .
C                                      .Y
C
C     DEFINITIONS
C     ANGLES (ENTERED IN DEGREES)
C     ALPHA  ROTATION OF U INTO V
C     BETA   ROTATION OF W INTO U
C     GAMMA  ROTATION OF V INTO W
C     PSI    ROTATION OF X INTO Y
C     THETA  ROTATION OF Z INTO X
C     PHI    ROTATION OF Y INTO Z
C     OBJMV  MOVEMENT OF UVW ORIGIN
C            MEASURED IN XYZ COORDS
C     VIEWMV MOVEMENT OF XYZ AXES
C     FOCUS  VIEWING DISTANCE FROM
C            PICTURE PLANE (YZ PLANE)
C     YOFF   HORIZONTAL OFFSET IN PICTURE PLANE
C     ZOFF   VERTICAL   OFFSET IN PICTURE PLANE
C     SCLE   NORMALIZED DISTANCE FROM CENTER TO EDGE OF PICT
C            NORMAL VALUE FOR FULL PICTURE IS 1/2
C     NORMAL RATIO OF SCLE TO FOCUS IS BETWEEN 1/4 AND 1/3
C     IF FOCAL LENGTH IS INCREASED WITH SCLE HELD CONSTANT
C     THE RESULT IS AS IF A TELEPHOTO LENS WERE USED
C     DECREASING THE FOCAL LENGTH CORRESPONDS TO USING A
C     WIDE ANGLE LENS
C     VIEWING ANGLES MEASURED VERTICALLY AND HORIZONTALLY
C     FROM CENTER ARE APPROXIMATELY 14 DEGREES FOR A RATIO
C     OF 1/4 AND 18.5 DEGREES FOR A RATIO OF 1/3
C
      CALL ROTATN(ROTMAT,ALPHA,BETA,GAMMA)
      DO 1 I=1,3
    1 ROTMAT(4,I)=OBJMV(I)-VIEWMV(I)
      A=-PSI
      B=-THETA
      C=-PHI
      CALL EXRTAT(ORIMAT,A,B,C)
      CALL PROD(ROTMAT,ORIMAT,4,4,4,HMATRX,4,4,4)
      FCTR1=-SCLE/FOCUS
      HMATRX(1,4)=HMATRX(1,1)*FCTR1
      HMATRX(2,4)=HMATRX(2,1)*FCTR1
      HMATRX(3,4)=HMATRX(3,1)*FCTR1
      HMATRX(4,4)=(HMATRX(4,1)-FOCUS)*FCTR1
      IF(YOFF.EQ.0.)GO TO 3
      FCTR2=YOFF/SCLE
      DO 2 I=1,4
    2 HMATRX(I,2)=HMATRX(I,2)+FCTR2*HMATRX(I,4)
    3 CONTINUE
      IF(ZOFF.EQ.0.)GO TO 5
      FCTR3=ZOFF/SCLE
      DO 4 I=1,4
    4 HMATRX(I,3)=HMATRX(I,3)+FCTR3*HMATRX(I,4)
    5 CONTINUE
      CALL ROTATN(ORIMAT,PSI,THETA,PHI)
      DO 6,I=1,3
      TEMP(I)=VIEWMV(I)-OBJMV(I)
      TEMPF(I)=ORIMAT(1,I)*FOCUS+TEMP(I)
      TEMPY(I)=ORIMAT(2,I)+TEMP(I)
    6 TEMPZ(I)=ORIMAT(3,I)+TEMP(I)
      A=-ALPHA
      B=-BETA
      C=-GAMMA
      CALL ROTATN(ROTMAT,A,B,C)
      DO 7 I=1,12
    7 OBSPOS(I)=0.0
      DO 8 J=1,3
      JJ=J+3
      JJJ=J+6
      J4J=J+9
      DO 8 I=1,3
      OBSPOS(JJ)=OBSPOS(JJ)+TEMPY(I)*ROTMAT(I,J)
      OBSPOS(JJJ)=OBSPOS(JJJ)+TEMPZ(I)*ROTMAT(I,J)
      OBSPOS(J4J)=OBSPOS(J4J)+TEMPF(I)*ROTMAT(I,J)
    8 OBSPOS(J)=OBSPOS(J)+TEMP(I)*ROTMAT(I,J)
      RETURN
      END


      SUBROUTINE FXRTAT(RMATRX,A,B,C)
      DIMENSION RMATRX(4,4)
C         THIS SUBROUTINE CALCULATES THE 4 X 4 ROTATION
C         MATRIX FOR EULER ANGLE ROTATION
C         CALCULATE DIRECTIONAL COSINES
C         AND PLACE IN ROTATION MATRIX
      TPI=3.1416/180.
      ALPHA=TPI*A
      BETA=TPI*B
      GAMMA=TPI*C
      SALPHA=SIN(ALPHA)
      CALPHA=COS(ALPHA)
      SBETA=SIN(BETA)
      CBETA=COS(BETA)
      SGAMMA=SIN(GAMMA)
      CGAMMA=COS(GAMMA)
C         CALCULATE DIRECTIONAL COSINES
C         AND PLACE IN ROTATION MATRIX
      TEMP1=CGAMMA*SALPHA
      TEMP2=SGAMMA*CALPHA
      TEMP3=SGAMMA*SALPHA
      TEMP4=CGAMMA*CALPHA
      RMATRX(1,1)=CBETA*CALPHA
      RMATRX(1,2)=TEMP1+TEMP2*SBETA
      RMATRX(1,3)=TEMP3-TEMP4*SBETA
      RMATRX(2,1)=-CBETA*SALPHA
      RMATRX(2,2)=TEMP4-TEMP3*SBETA
      RMATRX(2,3)=TEMP2+TEMP1*SBETA
      RMATRX(3,1)=SBETA
      RMATRX(3,2)=-CBETA*SGAMMA
      RMATRX(3,3)=CGAMMA*CBETA
      DO 1 I=1,3
      RMATRX(I,4)=0.
    1 RMATRX(4,I)=0.
      RMATRX(4,4)=1.
      RETURN
      END


      SUBROUTINE ROTATN(RMATRX,A,B,C)
      DIMENSION RMATRX(4,4)
C         THIS SUBROUTINE CALCULATES THE 4 X 4 ROTATION MATR
C         ANGLE ROTATION.
      TPI=3.1416/180.
      ALPHA=TPI*A
      BETA=TPI*B
      GAMMA=TPI*C
      SALPHA=SIN(ALPHA)
      CALPHA=COS(ALPHA)
      SBETA=SIN(BETA)
      CBETA=COS(BETA)
      SGAMMA=SIN(GAMMA)
      CGAMMA=COS(GAMMA)
C         CALCULATE DIRECTIONAL COSINES
C         AND PLACE IN ROTATION MATRIX
      TEMP1=CGAMMA*SALPHA
      TEMP2=SGAMMA*CALPHA
      TEMP3=SGAMMA*SALPHA
      TEMP4=CGAMMA*CALPHA
      RMATRX(1,1)=CBETA*CALPHA
      RMATRX(1,2)=CBETA*SALPHA
      RMATRX(1,3)=-SBETA
      RMATRX(2,1)=TEMP2*SBETA-TEMP1
      RMATRX(2,2)=TEMP3*SBETA+TEMP4
      RMATRX(2,3)=SGAMMA*CBETA
      RMATRX(3,1)=SBETA*TEMP4+TEMP3
      RMATRX(3,2)=SBETA*TEMP1-TEMP2
      RMATRX(3,3)=CBETA*CGAMMA
      DO 1 I=1,3
      RMATRX(I,4)=0.
    1 RMATRX(4,I)=0.
      RMATRX(4,4)=1.
      RETURN
      END


      SUBROUTINE PROD (A,B,N,M,L,C,ND,MD,LD)
      DIMENSION A(ND,MD),B(MD,LD),C(ND,LD)
      DO 1 I=1,ND
      DO 1 J=1,LD
    1 C(I,J)=0.
      DO 151 I=1,N
      DO 151 J=1,L
      DO 151 K=1,M
  151 C(I,J) = C(I,J) + A(I,K)*B(K,J)
      RETURN
      END


      SUBROUTINE DESCRB(ARRAY,IARRAY,FNORML,LSTFAC,N,SUBR1,
     1XSTART,YSTART,XEND,YEND,JDSPLY,LSTBFC,NPNTR)
      COMMON/SDFB/ALPHA,BETA,GAMMA,PSI,THETA,PHI,OBJMV(3),
     1VIEWMV(3),FOCUS,YOFF,ZOFF,SCLE,HMATRX(4,4),OBSPOS(12)
      DIMENSION ARRAY(N,13),IARRAY(2,N,13),LSTFAC(1)
      DIMENSION XSTART(1),YSTART(1),XEND(1),YEND(1)
      DIMENSION IWRKNR(20),JGSET(20),INTLVP(20)
      DIMENSION TEMPIN(9),FNORML(N,3),LSTBFC(1)
      DIMENSION AA(3),BB(3),CC(3),PP(3)
      DIMENSION AAA(3),BBB(3),AAAE(3),BBBE(3),TSTPNT(4)
      INTEGER CURSEG,TPNTR,POINT,TEMP
C
C         N IS THE NUMBER OF DIFFERENT POINTS EXPECTED
C         SUBR1 IS THE SUBROUTINE THAT WILL ENTER DATA
C         CALLING NAME OF SUBR1 MUST BE DECLARED EXTERNAL
C         IN THE MAIN CALLING PROGRAM IN ORDER THAT ITS
C         NAME WILL BE RECOGNIZED AS A SUBROUTINE
C         DATA IS ENTERED IN THE FIRST THREE LOCATIONS
C         OF TEMPIN
C         SUBR1 SHOULD READ IN NUMBER OF POINTS TO THE FACE
C         THIS SHOULD BE FOLLOWED BY THE THREE COORDINATES
C         OF EACH POINT, ONE POINT AT A TIME, INTO ARRAY
C         TEMPIN
C         IF DESIRED, READCR MAY BE USED TO ENTER DATA
C         FORMAT EXPECTED IS 1I10 FOR THE NUMBER OF POINTS
C         OF THAT FACE TO FOLLOW
C         SUBSEQUENT CARDS WILL USE THE FORMAT 3F20.6 FOR
C         COORDINATES U,V,W
C
```

```
      TRISNS(AO,BO,A1,B1,A2,B2,CONST)=AO*B1+A2*BO-AO*B2-A1*
     1BO+CONST
      NPNTR=LPNTR=IFPNTR=LSPNTR=JDSPLY=JPNTR=LRESTR=0
      ALPHA=BETA=GAMMA=PSI=THETA=PHI=0.
      VIEWMV(1)=VIEWMV(2)=VIEWMV(3)=0.
      OBJMV(2)=OBJMV(3)=0.
      OBJMV(1)=-40.
      FOCUS=1.5
      ZOFF=YOFF=0.
      SCLE=0.5
      DO 1 I=1,N
      ARRAY(I,7)=0.
    1 ARRAY(I,13)=0.
      RETURN


      SUBROUTINE ERASE
      DO 7 I=1,N
    7 ARRAY(I,13)=0.
      IF(LRESTR.EQ.0)GO TO 8
      DO 6 I=1,LPNTR
    6 IARRAY(2,I,9)=IABS(IARRAY(2,I,9))
      RETURN
    8 CONTINUE
      CALL LINES
      RETURN


      SUBROUTINE FACES
      NPT=1
      CALL SUBR1(NPT,TEMPIN)
      IFPNTR=IFPNTR+1
      IARRAY(1,IFPNTR,8)=NPT
      IARRAY(2,IFPNTR,8)=LSPNTR+1
      DO 13 IN=1,NPT
      CALL SUBR1(0,TEMPIN)
   21 DO 22 I=1,NPNTR
      IF(TEMPIN(1).EQ.ARRAY(I,1).AND.TEMPIN(2).EQ.ARRAY(I,2)
     1.AND.TEMPIN(3).EQ.ARRAY(I,3))GO TO 14
   22 CONTINUE
      NPNTR=NPNTR+1
      NEW=NPNTR
      DO 11 I=1,3
   11 ARRAY(NPNTR,I)=TEMPIN(I)
   12 LSPNTR=LSPNTR+1
   13 LSTFAC(LSPNTR)=NEW
      RETURN
   14 NEW=I
      GO TO 12


      SUBROUTINE LINES
      DO 40 I=1,IFPNTR
      IR=1
      NBR=IARRAY(1,I,8)
      TPNTR=IARRAY(2,I,8)
      LSTRT=TPNTR
      LPNT=LSTFAC(TPNTR)
   31 TPNTR=TPNTR+1
      NPNT=LSTFAC(TPNTR)
   32 IR=IR+1
      IF(NPNT.LT.LPNT)GO TO 38
      IB=1
      LLOW=LPNT
      LHIGH=NPNT
   39 CONTINUE
      DO 16 J=1,LPNTR
      IF(IARRAY(1,J,9).EQ.LLOW.AND.IARRAY(2,J,9).EQ.LHIGH)GO
     1TO 37
   16 CONTINUE
      LPNTR=LPNTR+1
      IF(LRESTR.NE.0)GO TO 33
      IARRAY(IB,LPNTR,11)=I
   33 IARRAY(1,LPNTR,9)=LLOW
      IARRAY(2,LPNTR,9)=LHIGH
   34 LPNT=NPNT
      IF(IP-NBR)31,36,40
   36 NPNT=LSTFAC(LSTRT)
      GO TO 32
   37 IARRAY(IB,J,11)=I
      GO TO 34
   38 LLOW=NPNT
      LHIGH=LPNT
      IB=2
      GO TO 39
   40 CONTINUE
      LRESTR=0
      RETURN


      SUBROUTINE CCVCVX
C         THE FIRST POINT ENTERED MUST BE A CORNER POINT
C          I.E., THE LAST, FIRST, AND SECOND POINTS MUST
C          NOT BE COLLINEAR
      DO 49 I=1,IFPNTR
      NBR=IARRAY(1,I,8)
      TPNTR=IARRAY(2,I,8)
      POINT=LSTFAC(TPNTR+NBR-1)
      DO 41 J=1,3
   41 TEMPIN(J)=ARRAY(POINT,J)
      POINT=LSTFAC(TPNTR)
      DO 42 J=1,3
   42 TEMPIN(J+3)=ARRAY(POINT,J)
      POINT=LSTFAC(TPNTR+1)
      DO 43 J=1,3
   43 TEMPIN(J+6)=ARRAY(POINT,J)
      A=SQRT((TEMPIN(1)-TEMPIN(4))**2+(TEMPIN(2)-TEMPIN(5))
     1**2+(TEMPIN(3)-TEMPIN(6))**2)
      B=SQRT((TEMPIN(7)-TEMPIN(4))**2+(TEMPIN(8)-TEMPIN(5))
     1**2+(TEMPIN(9)-TEMPIN(6))**2)
      ABI=1./(A*B)
      FNORML(I,1)=ABI*((TEMPIN(2)-TEMPIN(5))*(TEMPIN(9)-
     1TEMPIN(6))-(TEMPIN(3)-TEMPIN(6))*(TEMPIN(8)-
     2TEMPIN(5)))
      FNORML(I,2)=ABI*((TEMPIN(3)-TEMPIN(6))*(TEMPIN(7)-
     1TEMPIN(4))-(TEMPIN(1)-TEMPIN(4))*(TEMPIN(9)-
     2TEMPIN(6)))
      FNORML(I,3)=ABI*((TEMPIN(1)-TEMPIN(4))*(TEMPIN(8)-
     1TEMPIN(5))-(TEMPIN(2)-TEMPIN(5))*(TEMPIN(7)-
     2TEMPIN(4)))
   49 CONTINUE
      DO 47 I=1,LPNTR
      IFACE1=IABS(IARRAY(1,I,11))
      IFACE2=IARRAY(2,I,11)
      LLOW=IARRAY(1,I,9)
      LHIGH=IARRAY(2,I,9)
      DO 44 J=1,3
      AA(J)=FNORML(IFACE1,J)
   44 BB(J)=FNORML(IFACE2,J)
      CALL CROSS(AA,BB,CC)
      DO 45 J=1,3
   45 AA(J)=ARRAY(LHIGH,J)-ARRAY(LLOW,J)
      SCALAR=0.
      DO 46 J=1,3
   46 SCALAR=AA(J)*CC(J)+SCALAR
      IF(SCALAR.GE.0.)GO TO 47
      IARRAY(1,I,11)=-IFACE1
      IARRAY(1,LLOW,7)=1
      IARRAY(1,LHIGH,7)=1
   47 CONTINUE
      RETURN
      SUBROUTINE HDNSRF
      JPNTR=0
      DO 54 I=1,IFPNTR
      NBR=IABS(IARRAY(1,I,8))
      TPNTR=IARRAY(2,I,8)
      POINT=LSTFAC(TPNTR+NBR-1)
      DO 51 J=1,2
   51 TEMPIN(J)=ARRAY(POINT,J+4)
      POINT=LSTFAC(TPNTR)
      DO 52 J=3,4
   52 TEMPIN(J)=ARRAY(POINT,J+2)
      POINT=LSTFAC(TPNTR+1)
      DO 53 J=5,6
   53 TEMPIN(J)=ARRAY(POINT,J)
      DEPTHN=(TEMPIN(1)-TEMPIN(3))*(TEMPIN(6)-TEMPIN(4))-
     1(TEMPIN(5)-TEMPIN(3))*(TEMPIN(2)-TEMPIN(4))
      IF(DEPTHN.GE.0.)GO TO 55
      IARRAY(1,I,8)=NBR
   54 CONTINUE
      DO 58 I=1,LPNTR
      IFLAG1=0
      TEMP=IABS(IARRAY(1,I,9))
      IFACE1=IARRAY(1,I,11)
      IF(IFACE1.GT.0)GO TO 57
      IFACE1=-IFACE1
      IFLAG1=1
   57 IFACE2=IARRAY(2,I,11)
      IF(IARRAY(1,IFACE1,8).LT.0)IFLAG1=IFLAG1+1
      IF(IARRAY(1,IFACE2,8).LT.0)IFLAG1=IFLAG1+1
      IF(IFLAG1.GE.2)GO TO 56
      IARRAY(1,I,9)=TEMP
   58 CONTINUE
      RETURN
   55 IARRAY(1,I,8)=-NBR
      GO TO 54
   56 IARRAY(1,I,9)=-TEMP
      GO TO 58


      SUBROUTINE CMPUT
C         EXAMINE SEGMENTS ORIGINATING AT SPECIFIED NODE
      JDSPLY=0
      DO 104 ILK=1,LPNTR
      LLOW=IARRAY(1,ILK,9)
      IF(LLOW.LT.0)GO TO 104
      XLOW=ARRAY(LLOW,5)
      YLOW=ARRAY(LLOW,6)
      IF(IARRAY(2,LLOW,13).NE.0)GO TO 105
      CALL VRTNTR
  105 CONTINUE
C         SEARCH FIRST NODE OF ALL SEGMENTS
      DO 114 IL=ILK,LPNTR
      IF(IARRAY(1,IL,9).NE.LLOW)GO TO 114
      LHIGH=IARRAY(2,IL,9)
      IF(LHIGH.LT.0)GO TO 114
C         ENTER NATURE AT INITIAL VERTEX
  112 NBR=IARRAY(1,LLOW,13)
      TPNTR=IARRAY(2,LLOW,13)
      IWRKNR(1)=NBR
      IF(NBR.EQ.0)GO TO 301
      IDRAW=0
  107 IEND=NBR+1
      DO 111 J=2,IEND
  111 IWRKNR(J)=LSTBFC(TPNTR+J-2)
  113 CONTINUE
C         FIND DIRECTION NUMBERS
      XHIGH=ARRAY(LHIGH,5)
      YHIGH=ARRAY(LHIGH,6)
      AAL=XHIGH-XLOW
      BBL=YHIGH-YLOW
C           FIND LOSING INTERSECTIONS
      CUPSEG=IL
      CALL LNTRSC
      IF(INRSCT.EQ.0)GO TO 103
C           SEARCH FOR INTERSECTION AT INITIAL VERTEX
      ICOUNT=INRSCT
      IJK=LPNTR+1
      NTEMP=0
      DO 109 I=1,INRSCT
      IJK=IJK+1
      IF(IARRAY(1,IJK,9).LT.0)GO TO 109
      IF(XLOW.EQ.FNORML(I,1).AND.YLOW.EQ.FNORML(I,2))GO TO
     1109
C           INTERSECTION OCCURS AT INITIAL VERTEX
      NTEMP=NTEMP+1
      INTLVR(NTEMP)=IARRAY(1,IJK,9)
      IARRAY(1,IJK,9)=-IARRAY(1,IJK,9)
  109 CONTINUE
      IF(NTEMP.EQ.0)GO TO 151
      XNEAR=XLOW+.0001*AAL
      YNEAR=YLOW+.0001*BBL
C           COMPUTE DEPTH OF POINT NEAR VERTEX
      TEMPIN(1)=.001*ARRAY(LHIGH,1)-ARRAY(LLOW,1)*.999
      TEMPIN(2)=.001*ARRAY(LHIGH,2)-ARRAY(LLOW,2)*.999
      TEMPIN(3)=.001*ARRAY(LHIGH,3)-ARRAY(LLOW,3)*.999
      DPTHNR=DEPTH(TEMPIN(1),TEMPIN(2),TEMPIN(3))
      CALL TSTNNP
      IF(IDRAW.EQ.1.AND.IWRKNR(1).NE.0)GO TO 302
      IF(IDRAW.EQ.0.AND.IWRKNR(1).EQ.0)GO TO 303
      GO TO 151
  301 JDSPLY=JDSPLY+1
      XSTART(JDSPLY)=XLOW
      YSTART(JDSPLY)=YLOW
      IDRAW=1
      GO TO 113
  302 JDSPLY=JDSPLY-1
      IDRAW=0
      GO TO 151
  303 JDSPLY=JDSPLY+1
      XSTART(JDSPLY)=XLOW
      YSTART(JDSPLY)=YLOW
      IDRAW=1
  151 CONTINUE
      IF(ICOUNT.EQ.0) GO TO 103
      NTEMP=0
      IJK=LPNTR+1
      DO 152 I=1,INRSCT
      IJK=IJK+1
      IJKTST=IARRAY(1,IJK,9)
      IF(IJKTST.LT.0)GO TO 152
      IF(XHIGH.EQ.FNORML(I,1).AND.YHIGH.EQ.FNORML(I,2))GO
     1TO 153
  152 CONTINUE
      IFLAG=0
      IF(ICOUNT.LE.NTEMP)GO TO 201
      GO TO 154
  153 NTEMP=NTEMP+1
      INTLVR(NTEMP)=IJKTST
      IARRAY(1,IJK,9)=-IJKTST
      GO TO 152
  154 CONTINUE
      IJK=LPNTR+1
      RLNGTH=1.E6
      DO 155 I=1,INRSCT
      IJK=IJK+1
      IJKTST=IARRAY(1,IJK,9)
      IF(IJKTST.LT.0)GO TO 155
      ALSEG=FNORML(I,1)-XLOW
      BLSEG=FNORML(I,2)-YLOW
      TLNGTH=ABS(ALSEG)+ABS(BLSEG)
      IF(TLNGTH-RLNGTH)156,157,155
  156 RLNGTH=TLNGTH
      ISEGJ=IJKTST
      NXTINT=IJK
      MULTPL=1
      NI=I
      GO TO 155
  157 IF(IFLAG.EQ.0)MULTPL=MULTPL+1
  155 CONTINUE
      IFACE1=IABS(IARRAY(1,ISEGJ,11))
      IFACE2=IARRAY(2,ISEGJ,11)
      IF(MULTPL.GT.1)GO TO 162
  165 CONTINUE
      ITSTNR=IWRKNR(1)
      IF(IARRAY(1,IFACE1,8).LT.0)GO TO 160
      CALL DELETE(IWRKNR,IFACE1)
      IF(ITSTNR.NE.IWRKNR(1))GO TO 159
      IWRKNR(1)=IWRKNR(1)+1
      IWRKNR(ITSTNR+2)=IFACE1
      CALL DELETE(IWRKNR,IFACE2)
      GO TO 161
  159 IF(IARRAY(1,IFACE2,8).LT.0)GO TO 161
      IWRKNR(ITSTNR+1)=IFACE2
      IWRKNR(1)=IWRKNR(1)+1
      GO TO 161
  160 CALL DELETE(IWRKNR,IFACE2)
      IF(ITSTNR.NE.IWRKNR(1))GO TO 161
      IWRKNR(1)=IWRKNR(1)+1
      IWRKNR(ITSTNR+2)=IFACE2
  161 IARRAY(1,NXTINT,9)=-ISEGJ
      IF(IDRAW.EQ.1.AND.IWRKNR(1).NE.0)GO TO 304
      IF(IDRAW.EQ.0.AND.IWRKNR(1).EQ.0)GO TO 305
  168 ICOUNT=ICOUNT-1
      IF(ICOUNT.LE.NTEMP)GO TO 201
      MULTPL=MULTPL-1
      IF(MULTPL.LE.0)IFLAG=0
      GO TO 154
  304 XEND(JDSPLY)=FNORML(NI,1)
      YEND(JDSPLY)=FNORML(NI,2)
      CALL WINDOW
      IDRAW=0
      GO TO 168
  305 JDSPLY=JDSPLY+1
      XSTART(JDSPLY)=FNORML(NI,1)
      YSTART(JDSPLY)=FNORML(NI,2)
      IDRAW=1
      GO TO 168
  162 CONTINUE
C           MULTIPLE INTERSECTIONS
      IRPT=1
      ISCTND=IARRAY(1,ISEGJ,9)
      XINSCT=FNORML(NI,1)
      YINSCT=FNORML(NI,2)
  163 XSEGJ=ARRAY(ISCTND,5)
      YSEGJ=ARRAY(ISCTND,6)
      IF(XSEGJ.EQ.XINSCT.AND.YSEGJ.EQ.YINSCT)GO TO 166
      IF(IRPT.EQ.2)GO TO 165
  164 ISCTND=IABS(IARRAY(2,ISEGJ,9))
      IRPT=2
      GO TO 163
  166 IFLAG=1
      XLEFT=XINSCT-.001*AAL
      XRIGHT=XINSCT+.001*AAL
      YLEFT=YINSCT-.001*BBL
      YRIGHT=YINSCT+.001*BBL
  167 IF(IRPT.EQ.1)IHIGHE=IABS(IARRAY(2,ISEGJ,9))
      IF(IRPT.EQ.2)IHIGHE=IABS(IARRAY(1,ISEGJ,9))
      XTEST=XINSCT+.001*(ARRAY(IHIGHE,5)-XINSCT)
      YTEST=YINSCT+.001*(ARRAY(IHIGHE,6)-YINSCT)
      TEMP1=XLEFT*YRIGHT-XRIGHT*YLEFT
      AL=TRISNS(XTEST,YTEST,XLEFT,YLEFT,XRIGHT,YRIGHT,TEMP1)
      IF(ABS(AL).LT.1.0E-8)GO TO 170
      IF(AL.LT.0.)GO TO 161
      GO TO 165
  170 CONTINUE
      ITSTNR=IWRKNR(1)
      CALL DELETE(IWRKNR,IFACE1)
      IF(ITSTNR.NE.IWRKNR(1))GO TO 171
      IWRKNR(1)=ITSTNR+2
      IWRKNR(ITSTNR+2)=IFACE1
      IWRKNR(ITSTNR+3)=IFACE2
      GO TO 161
  171 CONTINUE
      CALL DELETE(IWRKNR,IFACE2)
      GO TO 161
  201 CONTINUE
      IF(NTEMP.EQ.0)GO TO 103
      XNEAR=XHIGH
      YNEAR=YHIGH
      DO 203 J=1,3
  203 TEMPIN(J)=ARRAY(LHIGH,J)
      DPTHNR=ARRAY(LHIGH,4)
      CALL TSTNNR
  103 CONTINUE
      IF(IDRAW.EQ.1)GO TO 306
  106 IF(IARRAY(2,LHIGH,13).NE.0)GO TO 204
      IARRAY(1,LHIGH,13)=IWRKNR(1)
      IARRAY(2,LHIGH,13)=JPNTR+1
      IEND=IWRKNR(1)+1
      DO 202 J=2,IEND
      JPNTR=JPNTR+1
      LSTBFC(JPNTR)=IWRKNR(J)
  202 CONTINUE
  204 IARRAY(2,IL,9)=-LHIGH
  114 CONTINUE
  104 CONTINUE
      RETURN
  306 XEND(JDSPLY)=XHIGH
      YEND(JDSPLY)=YHIGH
      CALL WINDOW
      GO TO 106


      SUBROUTINE ALTER
      NPNTRA=NPNTR
      DO 471 II=1,LPNTR
      MIRROR=1
      LLOW=IARRAY(1,II,9)
      IF(LLOW.LT.0)GO TO 471
      IF(ARRAY(LLOW,4).GT.0.)MIRROR=MIRROR+1
      LHIGH=IARRAY(2,II,9)
      IF(ARRAY(LHIGH,4).GT.0.)MIRROR=MIRROR+2
      GO TO (471,481,482,472),MIRROR
  472 IARRAY(1,II,9)=-LLOW
      GO TO 471
  481 LPOINT=LLOW
      HPOINT=LHIGH
      GO TO 483
  482 LPOINT=LHIGH
      HPOINT=LLOW
  483 CONTINUE
      DO 485 I=1,3
      AA(I)=ARRAY(LPOINT,I)
  485 BB(I)=ARRAY(HPOINT,I)
      CALL PIERC(OBSPOS(1),OBSPOS(4),OBSPOS(7),AA,BB,PP)
      NPNTRA=NPNTRA+1
      DO 486 I=1,3
      ARRAY(NPNTRA,I)=PP(I)
  486 ARRAY(NPNTRA,I+3)=HMATRX(4,I)
      FCTR=HMATRX(4,4)
      DO 487 I=1,3
      IJ=I+3
      FCTR=FCTR+ARRAY(NPNTR,I)*HMATRX(I,4)
      DO 487 J=1,3
  487 ARRAY(NPNTRA,IJ)=ARRAY(NPNTRA,IJ)+ARRAY(NPNTRA,J)*
     1HMATRX(J,I)
      DO 488 I=5,6
  488 ARRAY(NPNTRA,I)=ARRAY(NPNTRA,I)/FCTR
      MIRROR=MIRROR-1
      IARRAY(MIRROR,II,9)=NPNTRA
  471 CONTINUE
      IF(NPNTRA.NE.NPNTR)LRESTR=1
      RETURN


      SUBROUTINE VRTNTR
      MRPNTR=LPNTR+1
      ISTRT=MRPNTR+1
   86 NBR=0
      DO 92  IJ=1,LPNTR
      IFAC1=IABS(IARRAY(1,IJ,11))
      IFAC2=IARRAY(2,IJ,11)
      IF(IARRAY(1,IFAC1,8).LT.0.AND.IARRAY(1,IFAC2,8).LT.0.
     1OR.IFAC1.EQ.IFAC2) 0 TO 92
      LLOWE=IABS(IARRAY(1,IJ,9))
   98 LHIGHE=IABS(IARRAY(2,IJ,9))
      X1=ARRAY(LLOWE,5)
      X2=ARRAY(LHIGHE,5)
      XMAX=AMAX1(X1,X2)
      XMIN=AMIN1(X1,X2)
      IF(XMAX-XLOW)92,97,87
   87 IF(XMIN.GE.XLOW)GO TO 92
   88 Y1=ARRAY(LLOWE,6)
      Y2=ARRAY(LHIGHE,6)
      YMAX=AMAX1(Y1,Y2)
      IF(YMAX.LT.YLOW)GO TO 92
      TEMP1=X1*Y2-X2*Y1
      YMAX=YMAX+1.
      AL=TRISNS(XLOW,YLOW,X1,Y1,X2,Y2,TEMP1)
      AH=TRISNS(XLOW,YMAX,X1,Y1,X2,Y2,TEMP1)
      IF(AL.LT.0..AND.AH.LT.0..OR.AL.GT.0..AND.AH.GT.0.) GO
     1TO 92
C         MARK POLYGONS FOR INTERCEPT
      IEFAC1=IABS(IARRAY(1,IJ,11))
      IEFAC2=IARRAY(2,IJ,11)
      IF(IARRAY(1,IEFAC1,8).LT.0)GO TO 90
      DO 89  IK=ISTRT,MRPNTR
   89 IF(IARRAY(1,IK,11).EQ.IEFAC1)GO TO 95
      MRPNTR=MRPNTR+1
      IARRAY(1,MRPNTR,11)=IEFAC1
      IARRAY(2,MRPNTR,11)=1
   90 IF(IARRAY(1,IEFAC2,8).LT.0) GO TO 92
      DO 91  IK=ISTRT,MRPNTR
   91 IF(IARRAY(1,IK,11).EQ.IEFAC2)GO TO 96
      MRPNTR=MRPNTR+1
      IARRAY(1,MRPNTR,11)=IEFAC2
      IARRAY(2,MRPNTR,11)=1
   92 CONTINUE
      IARRAY(2,LLOW,13)=JPNTR+1
      DO 94  IJ=ISTRT,MRPNTR
      IF(IARRAY(2,IJ,11)-IARRAY(2,IJ,11)/2*2.EQ.0)GO TO 94
      IFACE=IARRAY(1,IJ,11)
      DO 93  J=1,3
   93 TEMPIN(J)=ARRAY(LLOW,J)
      CALL PLNLIN
      DVRTEX=ARRAY(LLOW,4)
      IF(ABS(DVRTEX).LT.0.1)GO TO 99
      PPX=DEPTH(PP(1),PP(2),PP(3))
      IF(ABS(PPX-DVRTEX).LT..01)GO TO 94
      IF(PPX.LT.DVRTEX.OR.PPX.GT.0.)GO TO 94
C         ENTER SET OF BLOCKING FACES
  101 JPNTR=JPNTR+1
      LSTBFC(JPNTR)=IARRAY(1,IJ,11)
      NBR=NBR+1
   94 CONTINUE
      IARRAY(1,LLOW,13)=NBR
      RETURN
   95 IARRAY(2,IK,11)=IARRAY(2,IK,11)+1
      GO TO 90
   96 IARRAY(2,IK,11)=IARRAY(2,IK,11)+1
      GO TO 92
   99 Y=ARRAY(LLOW,2)-OBSPOS(2)
      IF(ABS(Y).LT..1)GO TO 100
      IF(Y.GT.0..AND.PP(2).LT.0..OR.Y.LT.0..AND.PP(2).GT.0.)
     1GO TO 94
      IF(ABS(Y-PP(2)).LT..1)GO TO 94
      GO TO 101
  100 Z=ARRAY(LLOW,3)-OBSPOS(3)
      IF(ABS(Z).LT..1)GO TO 94
      IF(Z.GT.0..AND.PP(3).LT.0..OR.Z.LT.0..AND.PP(3).GT.0.)
     1GO TO 94
      IF(ABS(Z-PP(3)).LT..1)GO TO 94
      GO TO 101
   97 IF(XMAX.NE.XMIN)GO TO 88
      ARRAY(LLOWE,5)=ARRAY(LLOWE,5)-.00001
      ARRAY(LHIGHE,5)=ARRAY(LHIGHE,5)+.00001
      MRPNTR=LPNTR+1
      GO TO 86


      SUBROUTINE TSTNNP
C           TEST POINT NEAR TO KNOWN POINT
C           G IS THE SET OF ON VIEW FACES OF WHICH THE NODE
C           BEING TESTED IS A MEMBER
C           J IS THE SET OF ON VIEW FACES BELONGING TO THE
C           INTERSECTING SEGMENTS
C           FIND SET OF FACES WHICH MAKE UP J AND G
C
      JGSET(1)=0
  115 ISEGJ=INTLVR(NTEMP)
      IFACE1=IABS(IARRAY(1,ISEGJ,11))
      IFACE2=IARRAY(2,ISEGJ,11)
      IF(IARRAY(1,IFACE1,8).LT.0)GO TO 113
      CALL MERGE(JGSET,IFACE1)
  113 IF(IARRAY(1,IFACE2,8).LT.0) GO TO 114
      CALL MERGE(JGSET,IFACE2)
  114 NTEMP=NTEMP-1
      IF(NTEMP.GT.0)GO TO 115
      IEND=JGSET(1)+1
      DO 117 IT=2,IEND
      IFACE=JGSET(IT)
      NSEGS=IARRAY(1,IFACE,8)
      TPNTR=IARRAY(2,IFACE,8)
C           TEST NEAR POINT WITH RESPECT TO SETS J AND G
C           SEARCH FACE IFACE FOR INTERSECTION
  121 LLOWE=LSTFAC(TPNTR+NSEGS-1)
      LHIGHE=LSTFAC(TPNTR)
      ICROSS=0
      DO 122 IK=1,NSEGS
      X1=ARRAY(LLOWE,5)
      X2=ARRAY(LHIGHE,5)
      XMAX=AMAX1(X1,X2)
      XMIN=AMIN1(X1,X2)
      IF(XMAX-XNEAR)122,123,124
  124 IF(XMIN.GE.XNEAR)GO TO 122
  126 Y1=ARRAY(LLOWE,6)
      Y2=ARRAY(LHIGHE,6)
      YMAX=AMAX1(Y1,Y2)
      IF(YMAX.LT.YNEAR)GO TO 122
      TEMP1=X1*Y2-X2*Y1
      YMAX=YMAX+1
      AL=TRISNS(XNEAR,YNEAR,X1,Y1,X2,Y2,TEMP1)
      AH=TRISNS(XNEAR,YMAX,X1,Y1,X2,Y2,TEMP1)
      IF(AL.LT.0..AND.AH.GT.0..OR.AL.GT.0..AND.AH.LT.0.)
     1ICROSS=ICROSS+1
      LLOWE=LHIGHE
      LHIGHE=LSTFAC(TPNTR+IK)
  122 CONTINUE
```

```
      GO TO 116
  123 IF(XMAX.NE.XMIN)GO TO 126
      ARRAY(LLOWE,5)=ARRAY(LLOWE,5)-.00001
      ARRAY(LHIGHE,5)=ARRAY(LHIGHE,5)+.00001
      GO TO 121
  116 IF(ICROSS-ICROSS/2*2.NE.0)GO TO 118
      CALL DELETE(IWRKNR,IFACE)
      GO TO 120
  118 CONTINUE
C           POINT LIES INSIDE POLYGON-SEARCH SET
      IEND=IWRKNR(1)+1
      DO 119,IK=2,IEND
      IF(IFACE.EQ.IWRKNR(IK))GO TO 117
  119 CONTINUE
      CALL PLNLIN
      DPTHPP=DEPTH(PP(1),PP(2),PP(3))
      IF(DPTHPP.LT.DPTHNR) GO TO 117
      IWRKNR(IWRKNR(1)+2)=IFACE
      IWRKNR(1)=IWRKNR(1)+1
  120 ICOUNT=ICOUNT-1
  117 CONTINUE
      RETURN


      SUBROUTINE LNTRSC
C           THIS SUBROUTINE CALCULATES THE INTERSECTION OF ALL
C           SEGMENTS WITH A GIVEN SEGMENT IN THE PICTURE PLANE
      IXPNTR=LPNTR+1
      IOVRLP=INRSCT=JGSET(1)=0
      IFACE1=IABS(IARRAY(1,CURSEG,11))
      IFACE2=IARRAY(2,CURSEG,11)
      IF(ABS(AAL).LT..0001)GO TO 68
      CRNTSL=BBL/AAL
   61 TEMP2=YLOW-CRNTSL*XLOW
      TEMP3=XLOW*YHIGH-XHIGH*YLOW
      X1=ARRAY(LLOW,1)
      Y1=ARRAY(LLOW,2)
      Z1=ARRAY(LLOW,3)
      X2=ARRAY(LHIGH,1)
      Y2=ARRAY(LHIGH,2)
      Z2=ARRAY(LHIGH,3)
      TEMP1=X1*Y2-X2*Y1
      TEMP5=Y1*Z2-Y2*Z1
      TEMP6=Z1*X2-Z2*X1
      DO 702 I=1,3
      AAA(I)=ARRAY(LLOW,I)
  702 BBB(I)=ARRAY(LHIGH,I)
      DO 67 IEXAM=1,LPNTR
      IFAC1=IABS(IARRAY(1,IEXAM,11))
      IFAC2=IARRAY(2,IEXAM,11)
      IF(IFAC1.EQ.IFACE1.OR.IFAC2.EQ.IFACE1.OR.IFAC1.EQ.
     1IFACE2.OR.IFAC2.EQ.IFACE2.OR.IFAC1.EQ.IFAC2.OR.
     2IARRAY(1,IFAC1,8).LT.0.AND.IARRAY(1,IFAC2,8).LT.0)
     3GO TO 67
      LLOWE=IABS(IARRAY(1,IEXAM,9))
      IF(ARRAY(LLOWE,4).GT.1.)GO TO 67
      LHIGHE=IABS(IARRAY(2,IEXAM,9))
      XLOWE=ARRAY(LLOWE,5)
      YLOWE=ARRAY(LLOWE,6)
      XHIGHE=ARRAY(LHIGHE,5)
      YHIGHE=ARRAY(LHIGHE,6)
C          TEST FOR INTERSECTION USING TRI-SENSE TEST
      AL=TRISNS(XLOWE,YLOWE,XLOW,YLOW,XHIGH,YHIGH,TEMP3)
      AH=TRISNS(XHIGHE,YHIGHE,XLOW,YLOW,XHIGH,YHIGH,TEMP3)
C           ARE POINTS COLLINEAR
      INDAB1=1
      IF(ABS(AL).LT.1.E-5)INDAB1=INDAB1+1
      IF(ABS(AH).LT.1.E-5)INDAB1=INDAB1+2
      GO TO (611,613,614,72),INDAB1
  613 ICOMN=LLOWE
      IDIFF=LHIGHE
      GO TO 612
  614 ICOMN=LHIGHE
      IDIFF=LLOWE
      GO TO 612
  611 IF(AL.LT.0..AND.AH.LT.0..OR.AL.GT.0..AND.AH.GT.0.)
     1GO TO 67
  612 TEMP4=XLOWE*YHIGHE-XHIGHE*YLOWE
      AL=TRISNS(XLOW,YLOW,XLOWE,YLOWE,XHIGHE,YHIGHE,TEMP4)
      AH=TRISNS(XHIGH,YHIGH,XLOWE,YLOWE,XHIGHE,YHIGHE,TEMP4)
      IF(INDAB1.NE.1)GO TO 615
      INDAB2=1
      IF(ABS(AL).LT.1.E-5)INDAB2=INDAB2+1
      IF(ABS(AH).LT.1.E-5)INDAB2=INDAB2+2
      GO TO (615,616,617),INDAB2
  616 HINSCT=XLOW
      VINSCT=YLOW
      GO TO 701
  617 HINSCT=XHIGH
      VINSCT=YHIGH
      GO TO 701
  615 IF(AL.LT.0..AND.AH.LT.0..OR.AL.GT.0..AND.AH.GT.0.)
     1GO TO 67
      IF(INDAB1.NE.1)GO TO 620
  701 CONTINUE
      DO 703 I=1,3
      AAAE(I)=ARRAY(LLOWE,I)
  703 BBBE(I)=ARRAY(LHIGHE,I)
      CALL PIERC(AAA,OBSPOS(10),BBB,AAAE,BBBE,PP)
      IXX=1
  704 GO TO (711,712,713,67),IXX
  711 AL=TRISNS(OBSPOS(10),OBSPOS(11),X1,Y1,X2,Y2,TEMP1)
      AH=TRISNS(PP(1),PP(2),X1,Y1,X2,Y2,TEMP1)
      GO TO 714
  712 AL=TRISNS(OBSPOS(11),OBSPOS(12),Y1,Z1,Y2,Z2,TEMP5)
      AH=TRISNS(PP(2),PP(3),Y1,Z1,Y2,Z2,TEMP5)
      GO TO 714
  713 AL=TRISNS(OBSPOS(12),OBSPOS(10),Z1,X1,Z2,X2,TEMP6)
      AH=TRISNS(PP(3),PP(1),Z1,X1,Z2,X2,TEMP6)
  714 IXX=IXX+1
      IF(ABS(AL).LT.FOCUS)GO TO 704
      IF(AL.LT.0..AND.AH.GT.0..OR.AL.GT.0..AND.AH.LT.0.)GO
     1TO 67
      IF(INDAB2.NE.1)GO TO 64
  626 AALE=XHIGHE-XLOWE
      BBLE=YHIGHE-YLOWE
      IF(ABS(AALE).LT..0001)GO TO 69
      EXAMSL=BBLE/AALE
   62 HINSCT=(TEMP2+EXAMSL*XLOWE-YLOWE)/(EXAMSL-CRNTSL)
   63 VINSCT=TEMP2+CRNTSL*HINSCT
   64 CONTINUE
      IXPNTR=IXPNTR+1
      INRSCT=INRSCT+1
      FNORML(INRSCT,1)=HINSCT
      FNORML(INRSCT,2)=VINSCT
      IARRAY(1,IXPNTR,9)=IEXAM
      IF(IOVRLP) 81,67,76
  620 IF(JGSET(1).EQ.0)GO TO 622
      JEND=JGSET(1)+1
      DO 621 ISR=2,JEND
      IF(ICOMN.EQ.JGSET(ISR))GO TO 625
  621 CONTINUE
  622 JGSET(1)=JGSET(1)+1
      JGSET(JGSET(1)+1)=ICOMN
      INTLVR(JGSET(1))=IDIFF
      GO TO 67
  625 XO=ARRAY(IDIFF,5)
      YO=ARRAY(IDIFF,6)
      MARK=INTLVR(ISR-1)
      XT=ARRAY(MARK,5)
      YT=ARRAY(MARK,6)
      AL=TRISNS(XO,YO,XLOW,YLOW,XHIGH,YHIGH,TEMP3)
      AH=TRISNS(XT,YT,XLOW,YLOW,XHIGH,YHIGH,TEMP3)
      IF(AL.LT.0..AND.AH.LT.0..OR.AL.GT.0..AND.AH.GT.0.)GO
     1TO 621
      GO TO 701
   68 CRNTSL=1000.
      GO TO 61
   69 EXAMSL=1000.
      GO TO 62
C             OVERLAP HAS BEEN ESTABLISHED
C             CHECK FOR END POINTS OF COMMON SEGMENT
   72 IF(CRNTSL.GT.1000.)GO TO 77
      RIGHT=AMIN1(AMAX1(XLOW,XHIGH),AMAX1(XLOWE,XHIGHE))
      ALEFT=AMAX1(AMIN1(XLOW,XHIGH),AMIN1(XLOWE,XHIGHE))
      IF(RIGHT-ALEFT)67,74,75
   75 IOVRLP=1
   74 HINSCT=RIGHT
      GO TO 63
   76 HINSCT=ALEFT
      IOVRLP=0
      GO TO 63
C             SLOPE IS VERY HIGH USE VERTICAL COORDINATES
   77 RIGHT=AMIN1(AMAX1(YLOW,YHIGH),AMAX1(YLOWE,YHIGHE))
      ALEFT=AMAX1(AMIN1(YLOW,YHIGH),AMIN1(YLOWE,YHIGHE))
      IF(RIGHT-ALEFT)67,79,80
   80 IOVRLP=-1
   79 VINSCT=RIGHT
      HINSCT=XLOW
      GO TO 64
   81 VINSCT=ALEFT
      IOVRLP=0
      GO TO 64
   67 CONTINUE
      RETURN


      SUBROUTINE WINDOW
      IX1=XSTART(JDSPLY)
      IX2=XEND(JDSPLY)
      IY1=YSTART(JDSPLY)
      IY2=YEND(JDSPLY)
      IF(IX1.GE.1.AND.IX2.GE.1)GO TO 461
      IF(IX1.LE.-1.AND.IX2.LE.-1)GO TO 461
      IF(IY1.GE.1.AND.IY2.GE.1)GO TO 461
      IF(IY1.LE.-1.AND.IY2.LE.-1)GO TO 461
      IAGN=1
      X=XSTART(JDSPLY)
      Y=YSTART(JDSPLY)
      IX=IX1
      IY=IY1
  401 IF(IX)411,412,413
  411 VINSCT=Y+CRNTSL*(-1.-X)
  421 IF(ABS(VINSCT).LE.1.)GO TO 426
  423 IF(IY)424,461,425
  426 XTEMP=-1.
      GO TO 447
  424 HINSCT=X+(-1.-Y)/CRNTSL
      YTEMP=-1.
  422 IF(ABS(HINSCT).GT.1.)GO TO 461
      XTEMP=HINSCT
      GO TO 450
  425 HINSCT=X+(1.-Y)/CRNTSL
      YTEMP=1.
      GO TO 422
  412 IF(IY)424,432,425
  432 XTEMP=X
      YTEMP=Y
      GO TO 450
  413 VINSCT=Y+CRNTSL*(1.-X)
      IF(ABS(VINSCT).LE.1.)GO TO 446
      GO TO 423
  446 XTEMP=1.
  447 YTEMP=VINSCT
  450 GO TO (451,460),IAGN
  451 X=XEND(JDSPLY)
```

```
      Y=YEND(JDSPLY)
      IX=IX2
      IY=IY2
      IAGN=2
      XSTART(JDSPLY)=XTEMP
      YSTART(JDSPLY)=YTEMP
      GO TO 401
  460 CONTINUE
      XEND(JDSPLY)=XTEMP
      YEND(JDSPLY)=YTEMP
      RETURN
  461 JDSPLY=JDSPLY-1
      RETURN


      SUBROUTINE PLNLIN
      NTTL=IARRAY(1,IFACE,8)
      LOC=IARRAY(2,IFACE,8)
      POINT=LSTFAC(LOC+NTTL-1)
      DO 98  J=1,3
   98 AA(J)=ARRAY(POINT,J)
      POINT=LSTFAC(LOC)
      DO 99  J=1,3
   99 BB(J)=ARRAY(POINT,J)
      POINT=LSTFAC(LOC+1)
      DO 100 J=1,3
  100 CC(J)=ARRAY(POINT,J)
      CALL PIERC(AA,BB,CC,OBSPOS,TEMPIN,PP)
      RETURN


      SUBROUTINE DATSTR
      WRITE(6,10)
      DO 501 NPT=1,N
      WRITE(6,500)NPT,(ARRAY(NPT,J),J=1,6),((IARRAY(I,NPT,7)
     1,I=1,2)),((IARRAY(I,NPT,J),I=1,2),J=8,13)
  501 CONTINUE
      WRITE(6,20)
      WRITE(6,110)(LSTFAC(I),I=1,LSPNTR)
      WRITE(6,20)
      WRITE(6,110)(LSTBFC(I),I=1,JPNTR)
      DO 980 J=1,JDSPLY
      WRITE(6,990)XSTART(J),YSTART(J),XEND(J),YEND(J)
  980 CONTINUE
   10 FORMAT(1H1)
   20 FORMAT(///)
  110 FORMAT(5X,5I8,5X,5I8)
  500 FORMAT(1X,I3,2X,4F10.3,2F10.6,7(2X,2I4))
  990 FORMAT(5X,2F10.6,10X,2F10.6)
      RETURN
      END


      SUBROUTINE PIERC(P1,P2,P3,PL1,PL2,PP)
      DIMENSION P1(3),P2(3),P3(3),PL1(3),PL2(3),PP(3)
C        THIS SUBROUTINE CALCULATES THE INTERSECTION OF A
C         PLANE DETERMINED BY POINTS P1,P2,P3 AND A LINE
C         WHICH CONTAINS POINTS PL1 AND PL2.
C         THE COORDINATES OF THE INTERSECTION ARE RETURNED
C         THE ARRAY PP (PIERCING POINT)
C         THE DIRECTION NUMBERS OF THE NORMAL TO THE PLANE
C         ARE A, B, C.
      A=(P2(2)-P1(2))*(P3(3)-P1(3))-(P2(3)-P1(3))*(P3(2)-
     1P1(2))
      B=(P2(3)-P1(3))*(P3(1)-P1(1))-(P2(1)-P1(1))*(P3(3)-
     1P1(3))
      C=(P2(1)-P1(1))*(P3(2)-P1(2))-(P2(2)-P1(2))*(P3(1)-
     1P1(1))
      D=-A*P1(1)-B*P1(2)-C*P1(3)
      AL=PL2(1)-PL1(1)
      BL=PL2(2)-PL1(2)
      CL=PL2(3)-PL1(3)
      T=-(A*PL1(1)+B*PL1(2)+C*PL1(3)+D)/(A*AL+B*BL+C*CL)
      PP(1)=AL*T+PL1(1)
      PP(2)=BL*T+PL1(2)
      PP(3)=CL*T+PL1(3)
      RETURN
      END


      FUNCTION DEPTH(X,Y,Z)
      COMMON/SDFB/ALPHA,BETA,GAMMA,PSI,THETA,PHI,OBJMV(3),
     1VIEWMV(3),FOCUS,YOFF,ZOFF,SCLE,HMATRX(4,4),OBSPOS(12)
      DEPTH=X*HMATRX(1,1)+Y*HMATRX(2,1)+Z*HMATRX(3,1)+
     1HMATRX(4,1)
      RETURN
      END


      SUBROUTINE DELETE(LIST,ITEM)
      DIMENSION LIST(1)
      N=LIST(1)
      IEND=N+1
      DO 1 I=2,IEND
      IF(ITEM.EQ.LIST(I))GO TO 2
    1 CONTINUE
      RETURN
    2 IF(I.EQ.IEND)GO TO 3
      LIST(I)=LIST(IEND)
    3 LIST(1)=N-1
      RETURN
      END


      SUBROUTINE MERGE(LIST,ITEM)
      DIMENSION LIST(1)
      N=LIST(1)
      IEND=N+1
      DO 1 I=2,IEND
      IF(ITEM.EQ.LIST(I))GO TO 2
    1 CONTINUE
      LIST(N+2)=ITEM
      LIST(1)=N+1
    2 CONTINUE
      RETURN
      END


      SUBROUTINE READCR(N,TEMPIN)
      DIMENSION TEMPIN(3)
      IF(N.EQ.0)GO TO 1
      READ(5,2)N
      RETURN
    1 READ(5,3)(TEMPIN(I),I=1,3)
      RETURN
    2 FORMAT(1I10)
    3 FORMAT(3F20.6)
      END


      SUBROUTINE CROSS(A,B,C)
      DIMENSION A(3),B(3),C(3)
C           THIS SUBROUTINE CALCULATES THE CROSS PRODUCT
C           A X B FOR 3 DIMENSIONAL VECTORS
      C(1)=A(2)*B(3)-A(3)*B(2)
      C(2)=A(3)*B(1)-A(1)*B(3)
      C(3)=A(1)*B(2)-A(2)*B(1)
      RETURN
      END
      SUBROUTINE LABEL
      COMMON/SDFB/ALPHA,BETA,GAMMA,PSI,THETA,PHI,OBJMV(3),
     1VIEWMV(3),FOCUS,YOFF,ZOFF,SCLE,HMATRX(4,4),OBSPOS(12)
      WRITE(6,1)ALPHA,BETA,GAMMA,PSI,THETA,PHI,(OBJMV(I),
     1I=1,3),(VIEWMV(I),I=1,3),YOFF,ZOFF,FOCUS,SCLE
    1 FORMAT(4X,' ALPHA=',1F12.6,3X,'  BETA=',1F12.6,9X,
     1'GAMMA=',1F12.6,/7X,'PSI=',1F12.6,8X,' THETA=',1F12.6,
     28X,'    PHI=',1F12.6,/2X,'OBJMV(1)=',1F12.6,6X,
     3'OBJMV(2)=',1F12.6,6X,'OBJMV(3)=',1F12.6,/1X,
     4'VIEWMV(1)=',1F12.6,5X,'VIEWMV(2)=',1F12.6,5X,
     5'VIEWMV(3)=',1F12.6,/3X,'YOFF=',1F12.6,5X,'  ZOFF=',
     61F12.6,5X,'FOCUS=',1F12.6,5X,'SCALE=',1F12.6)
      RETURN
      END


      SUBROUTINE DRAWVP(NL,X1,Y1,X2,Y2)
      DIMENSION X1(1),Y1(1),X2(1),Y2(1)
      DIMENSION DRAW(10,200),JXY(200)
C         COLUMN DIMENSION OF DRAW AND DIMENSION OF JXY MUST
C         BE GREATER THAN OR EQUAL TO TWO TIMES THE NUMBER
C         OF LINES TO BE DRAWN
C         ISTEP IS THE NUMBER OF SYMBOLS PER LINE
C         ND IS THE ROW DIMENSION OF ARRAY DRAW
      ISTEP=10
      ND=10
      DO 1 J=1,NL
      JJ=2*J
      DRAW(1,JJ-1)=X1(J)
      DRAW(1,JJ)=Y1(J)
      XSTEP=(X2(J)-X1(J))/ISTEP
      YSTEP=(Y2(J)-Y1(J))/ISTEP
      DO 1 I=2,ISTEP
      DRAW(I,JJ-1)=DRAW(I-1,JJ-1)+XSTEP
      DRAW(I,JJ)=DRAW(I-1,JJ)+YSTEP
    1 CONTINUE
      NLTT=NL*2
      DO 2 I=1,NLTT
    2 JXY(I)=I
      CALL VPLOT(DRAW,JXY,ISTEP,ND,NL,1,-1.,1.,-1.,1.)
      RETURN
      END
```

BIBLIOGRAPHY

1. Anderson, G. B., Bertram, K. R., Conn, R. W., Malonquist, K. O., Millstein, R. E., and Tokulos, S., "Design of a Time-Sharing System Allowing Interactive Graphics, "Proceedings of the 23rd ACM National Conference, p. 1-6, 1968.

2. Appel, A., "The Notion of Quantitative Invisibility and the Machine Rendering of Solids," Proceedings of the 22nd ACM National Conference, p. 387-393, 1967.

3. Appel, A., "Some Techniques for Shading Machine Renderings of Solids," Proceedings of the Spring Joint Computer Conference, v. 32, p. 37-45, 1968.

4. Barlett, W. S., Busch, K. J., Flynn, M. L., and Salmon, R. L., "SIGHT, a Satellite Interactive Graphic Terminal," Proceedings of the 23rd National Conference, p. 499-509, 1968.

5. Boehm, B. W., "Tabular Representation of Multivariate Functions--With Applications to Topographic Modeling," Proceedings of the 22nd ACM National Conference, p. 403-415, 1967.

6. Brewer, S., "Data Base or Data Maze? An Exploration of Entry Points," Proceedings of the 23rd ACM National Conference, p. 623-630, 1968.

7. Brian, W. J., "A Parts Breakdown Technique Using List Structures," Communications of the ACM, v. 7, p. 362-365, June 1964.

8. Comba, P. G., "A Procedure for Detecting Intersections of Three-Dimensional Objects," Journal of the ACM, v. 15, p. 354-366, July 1968.

9. Comfort, W. T., "Multiword List Items," Communications of the ACM, v. 7, p. 357-362, June 1964.

10. Cotton, I. W. and Greatorex, J. R., "Data Structures and Techniques for Remote Computer Graphics," Proceedings of the Fall Joint Computer Conference, v. 33, p. 533-544, 1968.

11. Dertouzos, M. L. and Graham, H. L., "A Parametric Graphical Display Technique for On-Line Use," Proceedings of the Fall Joint Computer Conference, v. 29, p. 201-209, 1966.

12. Dertouzos, M. L., "PHASEPLOT: An On-Line Graphical Display Technique," IEEE Transactions of Electronic Computers, v. EC-16, p. 203-209, April 1967.

13. Erickson, W. L. and Soller, T. M., "Computer-Driven Display Systems," IEEE International Convention Record, part 3, p. 72-84, 1965.

14. Freeman, H., "On the Encoding of Arbitrary Geometric Configurations," IRE Transactions of Electronic Computers, v. EC-10, p. 260-268, June 1961.

15. Freeman, H., "Techniques for the Digital Computer Analysis of Chain-Encoded Arbitrary Plane Curves," Proceedings of the National Electronics Conference, p. 421-432, 1961.

16. Freeman, H. and Morse, S. P., "On Searching a Contour Map for a Given Terrain Elevation Profile," Journal of the Franklin Institute, v. 284, p. 1-25, July 1967.

17. Freeman, H. and Loutrel, P. P.,"An Algorithm for the Solution of the Two-Dimensional 'Hidden-Line' Problem," IEEE Transactions on Electronic Computers, v. EC-16, p. 784-790, December 1967.

18. Galimberti, R. and Montanari, U., "An Algorithm for Hidden Line Elimination," Communications of the ACM, v. 12, p. 206-211, April 1969.

19. Gray, J. C., "Compound Data Structure for Computer Aided Design; A Survey," Proceedings of the 22nd ACM National Meeting, p. 355-365, 1967.

20. Hagan, T. G. and Treiber, R., "Hybrid Analog/Digital Techniques for Signal Processing Applications," Proceedings of the Spring Joint Computer Conference, v. 28, p. 379-388, 1966.

21. Hagan, T. G., Nixon, R. J., and Schaefer, L. J., "The Adage Graphics Terminal," Proceedings of the Fall Joint Computer Conference, v. 33, p. 747-755, 1968.

22. Hurwitz, A., Citron, J. P., and Yeaton, J. B., "GRAF: Graphic Additions to FORTRAN," Proceedings of the Spring Joint Computer Conference, v. 30, p. 553-557, 1967.

23. Johnson, T. E., "A Computer Program for Drawing in Three Dimensions," Proceedings of the Spring Joint Computer Conference, v. 22, p. 347-353, 1963.

24. Kubert, B., Szabo, J., and Giulieri, S., "The Perspective Representation of Functions of Two Variables," Journal of the ACM, v. 15, p. 193-204, April 1968.

25. Kulsrud, H. E., "A General Purpose Graphic Language," Communications of the ACM, v. 11, p. 247-254, April 1968.

26. Long, C. A. and Gray, J. C., "ASP--A Ring Implemented Associative Structure Package," Communications of the ACM, v. 11, p. 550-555, August 1968.

27. Luh, J. Y. S. and Krolak, R. J., "A Mathematical Model for Mechanical Part Description," Communications of the ACM, v. 8, p. 125-129, February 1965.

28. Massachusetts Institute of Technology, Lincoln Lab, Report TR-296, *SKETCHPAD: A Man-Machine Graphical Communication System*, by I. E. Sutherland, January 1963.

29. Massachusetts Institute of Technology, Lincoln Lab, Report TR-315, *Machine Perception of Three-Dimensional Solids*, by L. G. Roberts, May 1963.

30. Massachusetts Institute of Technology, Lincoln Lab, Report TR-405, *On-Line Graphical Specification of Computer Procedures*, by W. R. Sutherland, 1966.

31. McLane, R. C. and Wolf, J. D., "Symbolic and Pictorial Displays for Submarine Control," *IEEE Transactions on Human Factors in Engineering*, v. HFE-8, p. 148-158, June 1967.

32. Morse, S. P., "A Mathematical Model for the Analysis of Contour-Line Data," *Journal of the ACM*, v. 15, p. 205-220, April 1968.

33. Morse, S. P., "Computer Storage of Contour-Map Data," *Proceedings of the 23rd ACM National Meeting*, p. 45-51, 1968.

34. Morse, S. P., "Concepts of Use in Contour Map Processing," *Communications of the ACM*, v. 12, p. 147-152, March 1969.

35. Naval Postgraduate School Computer Facility Technical Note No. 0211-03, *Plotting Package for IBM 360/67*, by P. C. Johnson, February 1969.

36. Newman, W. M., "A System for Interactive Graphical Programming," *Proceedings of the Spring Joint Computer Conference*, v. 32, p. 47-54, 1968.

37. Pfaltz, J. L. and Rosenfeld, A., "Computer Representation of Planar Regions by Their Skeletons," *Communications of the ACM*, v. 10, p. 119-125, February 1967.

38. Puckett, H. R., "Computer Method for Perspective Drawing," *Journal of Spacecraft and Rockets*, v. 1, p. 44-48, January 1964.

39. Rapkin, M. D. and Othman, M. A., "Stand-Alone/Remote Graphic System," *Proceedings of the Fall Joint Computer Conference*, v. 33, p. 731-746, 1968.

40. Roberts, L. G., *Graphical Communication and Control Languages*, Massachusetts Institute of Technology Lincoln Lab Reprint MS 1173, November 1964 (AD 626882).

41. Rome Air Development Center Report TR-67-310, *Promenade--An On-Line Pattern Recognition System*, by G. H. Ball and D. J. Hall at Stanford Research Institute, September 1967 (AD 822174).

42. Ross, D. T., "The AED Approach to Generalized Computer-Aided Design," *Proceedings of the 22nd ACM National Meeting*, p. 367-385, 1967.

43. Sproull, R. F. and Sutherland, I. E., "A Clipping Divider," Proceedings of the Fall Joint Computer Conference, v. 33, p. 765-775, 1968.

44. Sutherland, I. E., "A Head-Mounted Three Dimensional Display," Proceedings of the Fall Joint Computer Conference, v. 33, p. 757-764, 1968.

45. Thomas, E. M., "GRASP--A Graphical Service Program," Proceedings of the 22nd National Meeting, p. 395-402, 1967.

46. Tilton, H. B., "Principles of 3-D CRT Displays," Control Engineering, p. 74-78, February 1966.

47. Van Dam, A. and Evans, D., "A Compact Data Structure for Storing, Retrieving and Manipulating Line Drawings," Proceedings of the Spring Joint Computer Conference, v. 30, p. 601-610, 1967.

48. Vlahos, P., "The Three-Dimensional Display, Its Cues and Techniques," Information Display, p. 10-20, November/December 1965.

49. Weiss, R. A., "BE VISION, A Package of IBM 7090 FORTRAN Programs to Draw Orthographic Views of Combinations of Plane and Quadric Surfaces," Journal of the ACM, v. 13, p. 194-204, April 1966.

50. Wright Air Development Center Technical Note 55-747, Coordinate Systems for Solving the Three-Dimensional Flight Equations, by R. M. Howe at the University of Michigan, June 1956, (AD 111582).

51. Wylie, C., Romney, G., Evans, D., and Erdahl, A., "Half-Tone Perspective Drawings by Computer," Proceedings of the Fall Joint Computer Conference, v. 31, p. 49-58, 1967.

52. Zajac, E. E., "Computer-Made Perspective Movies as a Scientific and Communication Tool," Communications of the ACM, v. 7, p. 169-170, March 1964.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	20
2.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	Naval Ordnance Systems Command Department of the Navy Washington, D. C. 20360	1
4.	Associate Professor M. L. Cotton Code 52Cc Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	1
5.	Lieutenant Robert Bruce Desens, USN 468 Beacon Hill Circle Norfolk, Virginia 23502	1
6.	Naval Ordnance Systems Support Office, Altantic Bldg 62, Norfolk Naval Shipyard Portsmouth, Virginia 23709	1
7.	Computer Laboratory, Code 52Ec Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	1
8.	Computer Facility, Code 0211 Naval Postgraduate School Monterey, California 93940	1

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)*

Naval Postgraduate School
Monterey, California

2a. REPORT SECURITY CLASSIFICATION

Unclassified

2b. GROUP

3. REPORT TITLE

Computer Processing for Display of Three-Dimensional Structures

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

Electrical Engineer's Thesis; October 1969

5. AUTHOR(S) *(First name, middle initial, last name)*

Robert Bruce Desens

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
October 1969	236	52

8a. CONTRACT OR GRANT NO.

b. PROJECT NO.

c.

d.

9a. ORIGINATOR'S REPORT NUMBER(S)

9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*

10. DISTRIBUTION STATEMENT

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES

12. SPONSORING MILITARY ACTIVITY

Naval Postgraduate School
Monterey, California 93940

13. ABSTRACT

The field of computer graphics applied to three-dimensional space is introduced through a discussion of perspective transformations, data structure, contour lines, and the problem of hidden-line removal. The transformation of three-dimensional coordinates into two-dimensional picture-plane coordinates is developed for twelve degrees of freedom, allowing the simultaneous movement and rotation of both the object under view and the observer. Basic concepts and requirements for the structure of data and ideas for the use of contour lines are discussed as a relative part of the field of three-dimensional computer graphics. An algorithm for the removal of hidden lines is explained for the case where the objects under view can be assumed to be constructed of bounded plane surfaces.

DD FORM 1 NOV 65 1473 (PAGE 1)

S/N 0101-807-6811

Security Classification

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Perspective transformation						
Three-dimensional computer graphics						
Contour lines						
Hidden-line removal						
Graphics data structure						

DD FORM 1 NOV 65 1473 (BACK)

S/N 0101-807-6821

Security Classification

A-31409

thesD4503
Computer processing for display of three
3 2768 001 02816 0
DUDLEY KNOX LIBRARY